PL-TR-93-2211

SATELLITE IMAGERY AND TOPOGRAPHIC DATA IN VERIFICATION

David Simpson
Arthur Lerner-Lam

Lamont-Doherty Earth Observatory
of Columbia University
Route 9W, P.O. Box 1000
Palisades, NY 10964-8000

28 September 1993

Final Report
17 April 1989 - 30 June 1992

PHILLIPS LABORATORY
Directorate of Geophysics
AIR FORCE MATERIEL COMMAND
HANSCOM AIRFORCE BASE, MA 01731-3010

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Air Force or the U.S. Government.

This technical report has been reviewed and is approved for publication.

JAMES F. LEWKOWICZ
Program Manager

JAMES F. LEWKOWICZ, Chief
Solid Earth Geophysics Branch

DONALD H. ECKHARDT, Director
Earth Sciences Division

Qualified requestors may obtain additional copies from the Defense Technical Information Center.

If your address has changed, or if you wish to be removed from the mailing list, or if the addressee is no longer employed by your organization, please notify PL/TSI, 29 Randolph Road, Hanscom AFB MA 01731-3010. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	28 September 1993	Final: 17 Apr 1989-30 Jun 1992

4. TITLE AND SUBTITLE
Satellite Imagery and Topographic Data in Verification

5. FUNDING NUMBERS
F19628-89-K-0007
PE 62714E
PR 9A10
TA DA
WU AP

6. AUTHOR(S)
David Simpson
Arthur Lerner-Lam

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Lamont-Doherty Earth Observatory of Columbia University
Route 9W
Palisades, NY 10964

8. PERFORMING ORGANIZATION REPORT NUMBER
Final Report

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Phillips Laboratory
29 Randolph Road
Hanscom AFB, MA 01731-5000
Contract Manager: James Lewkowicz/GPEH

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
PL-TR-93-2211

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

We have developed analysis procedures, hardware platforms and software systems for the analysis of digital topographic data, satellite images, and generic time series. These have been implemented in a distributed processing environment on commonly available platforms and are available over Internet. Contact lerner@ldeo.columbia.edu for details.

Our imaging analysis methodologies have been applied to the southern border regions of the former Soviet Union. This area contains some of the most interesting tectonics on the planet and until recently has been relatively inaccessible to modern field programs. We demonstrate how views of available digital data sets can be combined to yield new insight and overviews of tectonic setting.

We have developed a spatial- and time-domain deconvolution methodology based on inverse theory, which offers the advantage of adaptation to noisy data and incorporation of scientific constraints. It is applied to the problem of singular deconvolution of noisy time series as an example.

14. SUBJECT TERMS
Digital topographic data; Satellite images; Generic time series.

15. NUMBER OF PAGES
126

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unclassified

TOPOGRAPHY AND SEISMICITY OF CENTRAL ASIA

David W. Simpson
Lamont-Doherty Earth Observatory
Palisades, NY
and
Gary L. Pavlis
Indiana University
Bloomington, IN

INTRODUCTION

The southern border of Asia is a unique tectonic environment on our planet. It contains the highest mountains on earth, the largest high plateau, and some of the highest deformation rates anywhere. It is now firmly established that the dominant force that has shaped the geography of central Asia is the collision of the Indian subcontinent and Eurasia which began in approximately mid-Tertiary times and has continued to the present [e.g. Sengor, 1979, 1984; Stocklin, 1989; Tapponier et al., 1981; and Zhang et al., 1984]

In spite of the unique setting of this area, large gaps exist in our knowledge of this region. A combination of political barriers and remote, difficult terrain has limited access to many relevant forms of data that could improve our understanding of the tectonics of this area. On the other hand, the past several years have borne witness to extraordinary changes in the political relationship between the western world and the republics that are the remnant of the former Soviet Union. These profound changes hold the promise of releasing a flood of new data relevant to the tectonics of this region. The prospect of improved access to existing data from the former Soviet Union, and prospects of increased levels of collaboration with scientists from the emerging new countries of this region is the driving force behind this paper.

Our focus is on the territory of the central Asian republics of the former Soviet Union including the mountain belts of the Pamir and western Tien Shan in the republics of Kazakhstan, Kirghistan, Tadjikistan, and Uzbekistan. This report consists of a series of annotated color maps of the seismicity and topography of Central Asia. The map area we use covers 62° - 82° E and 35° - 45° N. The topographic data are from the Defense Mapping Agency's Digital Terrain Elevation Data (DTED). Seismicity data are taken from the annual catalogs of "Earthquakes in the USSR" (ZSSSR). Starting in 1962, the Institute of Physics of the Earth of the Soviet Academy of Sciences began publication of annual catalogs of earthquakes, divided into various regions of the Soviet Union. Earthquakes as low as energy class 8 (~magnitude 2.2) are reported in the more recent catalogs, however coverage is not complete below class 9 (magnitude 3), except in some areas where special networks have been installed, such as the region around the Complex Seismological Expedition in Garm, Tadjikistan.

GEOGRAPHY

One of the most seismically active areas of the world is the Tien Shan- Pamir region of Soviet Central Asia. This region lies at the southern edge of the Asian continent, 400 km

north of the Indus suture. The Tien Shan mountains form a broad south-facing arc at the southern edge of the arid deserts of the southern Asian platform. The Pamir mountains form a tight north-facing arc at the north-western terminus of the Himalayan mountains. Within and between these major ranges lie basins of younger age - the Fergana Valley, Tadjik Depression, Tarim Basin and Issik Kul Basin.

The main physiographic regions can be seen in Figures 1 and 2. The Asian platform in this region is represented by the deserts (Kyzul Kum, Kara Kum and Moyun Kum) of the Turan Platform and the steppes of Kazakhstan. The main ranges of the northern Tien Shan are the E-W trending Alatau, the NW trending Talas Fergana and and the NE trending Chatkal. The main frontal range of the Southern Tien Shan extends as a broad south facing arc from the Kokshal Ranges in the east, through the Alai, to the Gissar, Turkistan and Zerafshan Ranges in the west. South of the Tien Shan, the tighter, north facing arc of the Pamir separates the Tadjik Depression from the Tarim Basin. A narrow inter-montaine valley, the Za-alai, divides the Tien Shan from the Pamir. The highest peaks of this part of Central Asia (Peak Communism 7,495 m and Peak Lenin, 7134 m) and are in the Akademi Nauk Ranges in the northern Pamir.

South of the border of the former Soviet Union, the region is bounded by the ranges of the Kopet Dag, Hindu Kush Karakorum and Kun Lun. Farther to the south lies the main India-Eurasian plate boundary.

The major rivers of Central Asia originate in the mountains of the Pamir and Tien Shan. The Hindu Kush presents the water divide between the south flowing Indus and the north flowing Amu Darya (Oxus). Both the Amu Darya and the Syr Darya (which has its headwaters in the northern Tien Shan) flow toward the Aral Sea; although little if any of the water from these sources actually enters the Aral since all is used up in extensive irrigation projects.

TOPOGRAPHY

Our starting point for this paper is digital topography data recently released for scientific use by the U.S. Defense Mapping Agency. These data were used to construct the base maps for this paper presented in Figures 1 and 2. The original digital data consisted of gridded topographic data in one degree cells with 1200 points per degree of longitude and latitude (approximately 90 m spacing NS and 70 m EW). We extracted the data from the area shown in Figure 1, merged these data into a common grid, and re-sampled the data along latitude lines, compressing the data by the cosine of the average latitude to give pseudo-Mercator projection. For the area shown in Figure 1 the full resolution topographic base thus contains (24000x12000) = 288×10^6 points. The full resolution pseudo-Mercator grid contains (18385x12000) 220×10^6 points. In order to provide a reasonable scale for plotting, the full resolution grid was smoothed and decimated by a factor of 8 to give a working grid of (2299x1500) = 3.4×10^6 points. We then produced the colored image that forms the base map in Figure 1 using the image processing technique described by Simpson and Anders (1992). First, a color scale was established in which the color of the image is directly related to absolute elevation. In particular, this image was constructed by smoothly interpolating between four basic colors: the lowest elevations are a deep blue color, intermediate elevations are a light green color, high elevations are brown, and the highest peaks are white. The second step in constructing this image was to calculate a

directional derivative map from the digital topography by convolving the topographic grid with a first difference operator. The derivative map is then used to modulate the color map by multiplying the three RGB (red, green, blue) color components for each point by the normalized gradient, so that bright areas represent positive slopes and dark areas represent negative slopes. This produces a texture on the image that can thought of as equivalent to sunlight shining at a low angle from the north west.

The shaded relief map clearly reveals some striking features related to the tectonics of this region. At scales of larger than 100 km the topography can be grouped into several major provinces. First, note four major areas that are relatively flat and topographically low: (1) the Kazakh platform; (2) the Tarim basin in China; (3) the Tadjik depression in Tadjikistan; and (4) the Turan platform which floors the Kyzyl Kum and most of the Kara Kum desert areas shown at the left of Figures 1 and 2. These level areas are adjacent to some of the highest mountains in the world that we would group into two major groups that we will refer to as the Himalayan and Tien Shan families. The "Himalayan family" is a collection of mountains forming the Hindu Kush, Pamir, Karakorum, and Kunlun mountains. These mountains are related because they form the western edge of the Himalayan mountain belt and the Tibetan plateau. The "Tien Shan family" is defined here as the series of mountains that lie between the Tadjik depression, Pamir mountains, and Tarim basins to the south; and the Kazakh platform to the north. The Tien Shan are tectonically distinct from the Himalayan family.

At scales of the order of the thickness of the lithosphere, we observe several first order features. The most striking of these is the large intermontain basin known as the Fergana Valley that separates the southern and northern sections of the Tien Shan. The nature of this basin is not very clear, but we note here the somewhat odd geometry of topographic features that surround this basin. The southern boundary of the Fergana Valley is in an east west direction parallel to the overall trend of the south Tien Shan. However, the northern edge of the basin is more triangular in shape as a result of a complicated transition between two different structural regimes. That is, the Chatkal Ranges at the northwestern boundary of the basin have a southwest- northeast oriented strike. The ridges forming the Chatkal ranges terminate abruptly against the Talas-Fergana fault. This is a strike-slip fault which can be seen easily in the topography as it runs at approximately N 45° W from the far western corner of the Tarim basin northward toward the Kyzyl Kum desert. This fault and structures that are probably related to it form the northeast boundary of the Fergana Valley.

The topography clearly delineate the well known arc shape of the Pamir running through Tadjikistan into China along the western boundary of the Tarim basin. There the arc of the Pamir merges into the western end of the Kunlun. To the south of this arc the topography rises to the extremely rugged interior of the Pamir and Karakorum mountains. The Tien Shan, in contrast to the Himalayan family of mountains, show considerably more variation in topographic "texture". Note that the Tien Shan are narrowest at the far eastern edge of this topographic image. As one moves to the west the Tien Shan widen to their maximum at about 75° E longitude where they intersect the Talas-Fergana fault. In this zone individual mountain ranges form the spines of a fan shaped assemblage that emanates from the mountains directly south of Lake Izyk Kul (the highest peaks in the Northern Tien Shan) at the eastern edge of this image, and terminates against the Talas Fergana fault.

There the strike of topographic ridges change dramatically to approximately N 45° E in the Chatkal ranges north of the Fergana Valley.

SEISMICITY

At least until 1992, the Institute of Physics of the Earth, Moscow issued an annual catalog of earthquakes in the USSR (Zemletriaseniia v SSSR, hereafter ZSSSR). These annual volumes contain articles and catalogs for different subregions of the USSR, based on seismic regionalization. The largest of these catalogs is for Central Asia. The area covered by the ZSSSR catalog extends beyond the borders of the USSR into Iran, Afghanistan and China to the south. Coverage of these border regions becomes less complete, however, because of the lack of stations in southern azimuths.

The increased coverage with regional and local stations results in a catalog that has a lower detection threshold and higher resolution than that of teleseismic catalogs such as the International Seismological Center (ISC). Figure 3 shows maps derived from the ISC and ZSSSR catalogs for the period 1962 - 1986 and the cumulative numbers of earthquakes reported in these catalogs is shown in Figures 4. The much greater resolution of the regional ZSSSR catalog, in terms of numbers of events, is clearly seen in Figure 4 which shows more than 5 times as many events reported as in the ISC catalog.

The catalog is based on data reported from stations only within the former USSR (see republic boundaries in Figure 2), so that coverage decreases to the south and especially the southeast. This explains the absence of events in the southeast compared to the ISC map. Epicenters in the catalog are reported to the nearest 0.1 or 0.01 degrees, resulting in the regular gridded pattern at tenth degree intervals for the less-well determined locations in the south and east.

In order to improve the presentation of the seismicity data and to assist in the comparison between topographic and tectonic features and the seismicity, Figures 7 to 9 show color representations of the seismicity, separated into crustal and subcrustal events and including a representation of event size.

Figure 7 shows all events except those for which the ZSSSR catalog depth is greater than 50 km and Figure 8 shows those for which the given depth is greater than 50 km. The representation of the of the sub-crustal, intermediate depth Hindu Kush - Pamir seismicity in Figure 8 is clear (Billington et al., 1977, Pavlis and Hamburger, 1991). However it is not clear that the events shown in Figure 7 are all crustal. A major problem with the ZSSSR catalog is the lack of resolution in depth determination. This is especially true for shallow events near the edge of the region covered by the regional networks. Many events are given either zero or no depth (i.e. blank) in the ZSSSR catalog. It is possible that some of these in the southern region near the Hindu Kush and Pamir are in fact mis-located sub-crustal events. In Figure 9, an attempt has been made to assist in overcoming this problem by accentuating only the larger of those events with questionable depth. One interesting feature of this map is the very low level of crustal activity beneath the high elevations of the northern Pamir. Another anomalous feature of the crustal seismicity of the Pamir is the cluster of normal faulting found in the south (Figure 11).

Many of the topographic features of Central Asia can be discerned in the patterns of seismicity in Figure 9. Among the more prominent features:

(a) There is intense activity along the southern front of the northern and southern Tien Shan. This zone, the Gissar Kokshal seismic zone, is the site of the largest earthquakes in Central Asia (Leith and Simpson, 1986) and marks the southern edge of what can be clearly identified as part of the "original" Asian plate.

(b) Relatively low levels of seismicity are found within the high mountainous areas. The Gissar and Kokshal Ranges of the Tien Shan are flanked on the south and north by high seismicity, but the higher elevations are much less active. Relatively low levels of activity in regions of high elevation are also found in the Talas Fergana Ranges and the highest section of the Pamir.

(c) Relatively low levels of seismicity are found within the major intermountain basins and depressions. Most prominent of these is the Issyk Kul Basin, where high activity along the ranges bordering the basin clearly outlines the relatively low seismicity of the basin itself. The Fergana Valley is similar but less obvious. The Tadjik Depression and Tarim Basin have high activity on their northern edges, near the Gissar Kokshal zone, but are less active in their interiors.

The seismicity is thus concentrated along the major topographic gradients dividing the distinct physiographic/tectonic provinces of the region, with relatively low activity in both high plateaus and low basins and higher activity along the flanks of the mountainous areas.

REFERENCES

Billington, S., B. L. Isacks, and M. Barazangi, Spatial distribution and focal mechanism of mantle earthquakes in the Hindu Kush-Pamir region: A contorted Benioff zone, Geology, 5, 699-704, 1977.

Leith, W.S. and D.W. Simpson, Seismic domains within the Gissar-Kokshal seismic zone, Soviet Central Asia, *J. Geophys. Res., 91*, 689-699, 1986

Pavlis, Gary L. and Michael W. Hamburger Aftershock sequences of intermediate depth earthquakes in the Pamir Hindu Kush seismic zone, J. Geophys Res. 96, 18107 - 18,117, 1991

Rautian, T. G., Earthquake energy (in Russian), in Methods of Detailed Investigations of Seismicity, U.V. Riznishenko, ed., Academy of Sciences USSR, Moscow, pp 75-113, 1960.

Sengor, A. M. C., The Cimmeride orogenic system and the tectonics of Eurasia, Geol. Soc. Amer., Special Paper 195, 75p., 1984.

Sengor, A. M. C., Mid-Mesozoic closure of the Permo-Triassic Tethys and its implications, Nature, 279, 590-593, 1979.

Simpson, D. and M. Anders, Tectonics and topography of the western United States - an application of digital mapping, *GSA Today*, 2, 117-121, 1992.

Stocklin, J., Tethys evolution in the Afghanistan-Pamir-Pakistan region, in Tectonic Evolution of the Tethyan Region, ed. A. M. C. Sengor, 241-264, 1989.

Tapponier, P.; M. Mattauer, F. Proust, and C. Cassaigneau, Mesozoic Ophiolites, sutures, and large-scale tectonic movements in Afghanistan, Earth and Planet. Sci. Lett., 52, 355-371.

Zhang, Z. M., J. R. Hou, R. G. Coleman, An outline of plate tectonics of China, Geol. Soc. Amer. Bull., 95, 295-312, 1984.

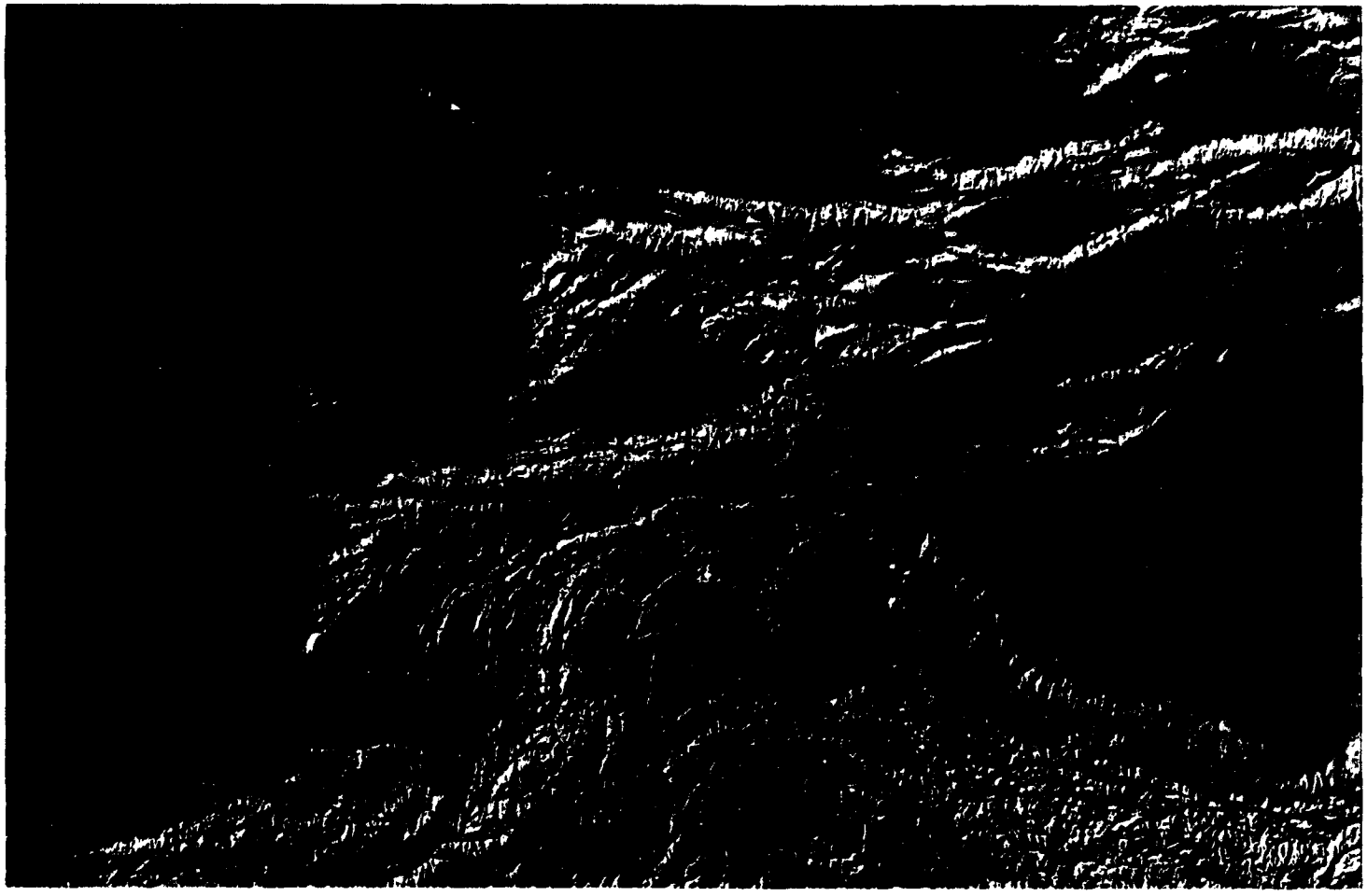


Figure 1: Shaded relief base map of the topography of Central Asia. Illumination is from the northwest. Projection is linear in latitude and longitude with the north-south scale adjusted to compress latitude by sin of average latitude. Shading is produced by multiplying each of the RGB color components of the elevation-dependent color at each point by the NW-SE gradient. The gradient at each point is determined by a simple difference operator.

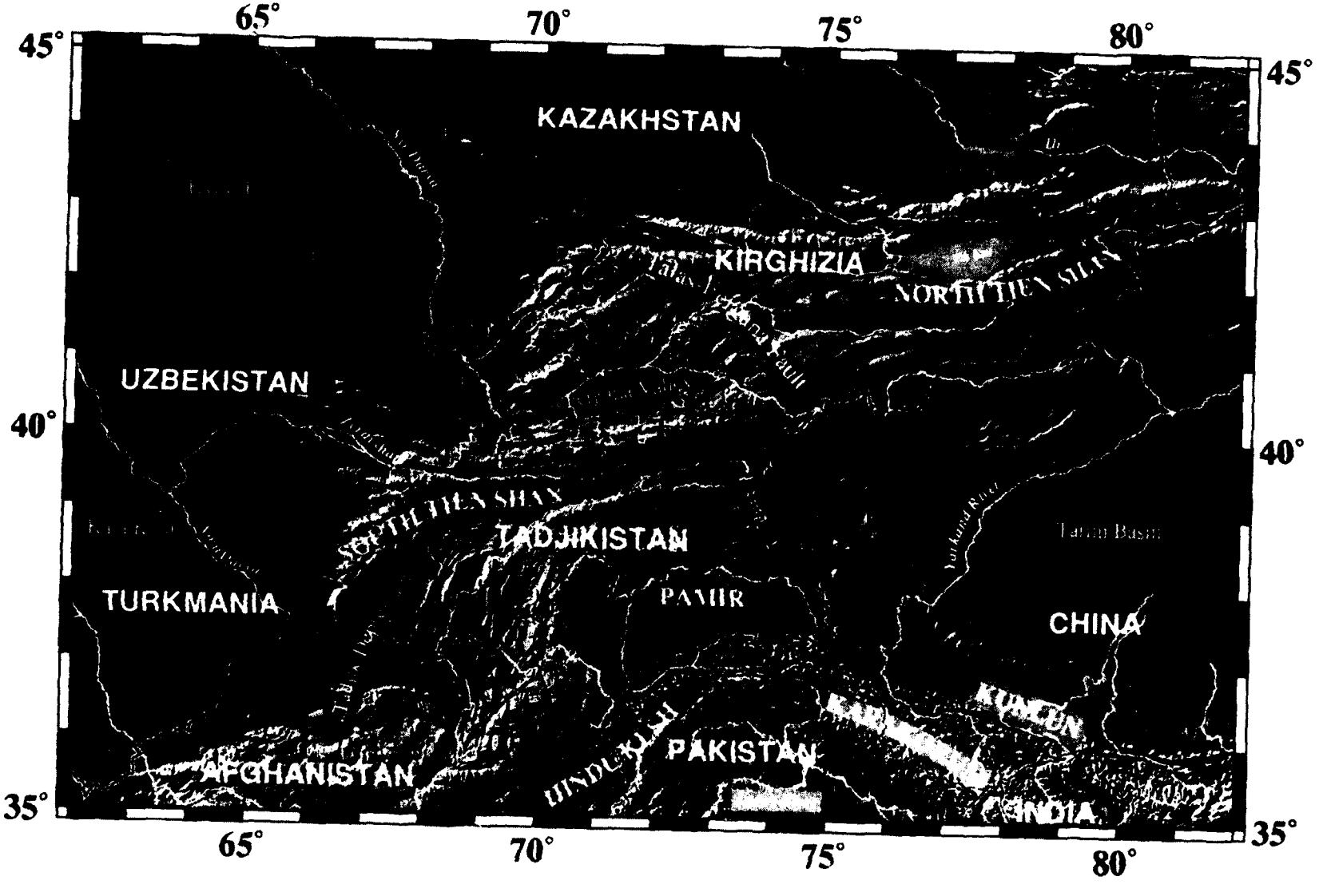
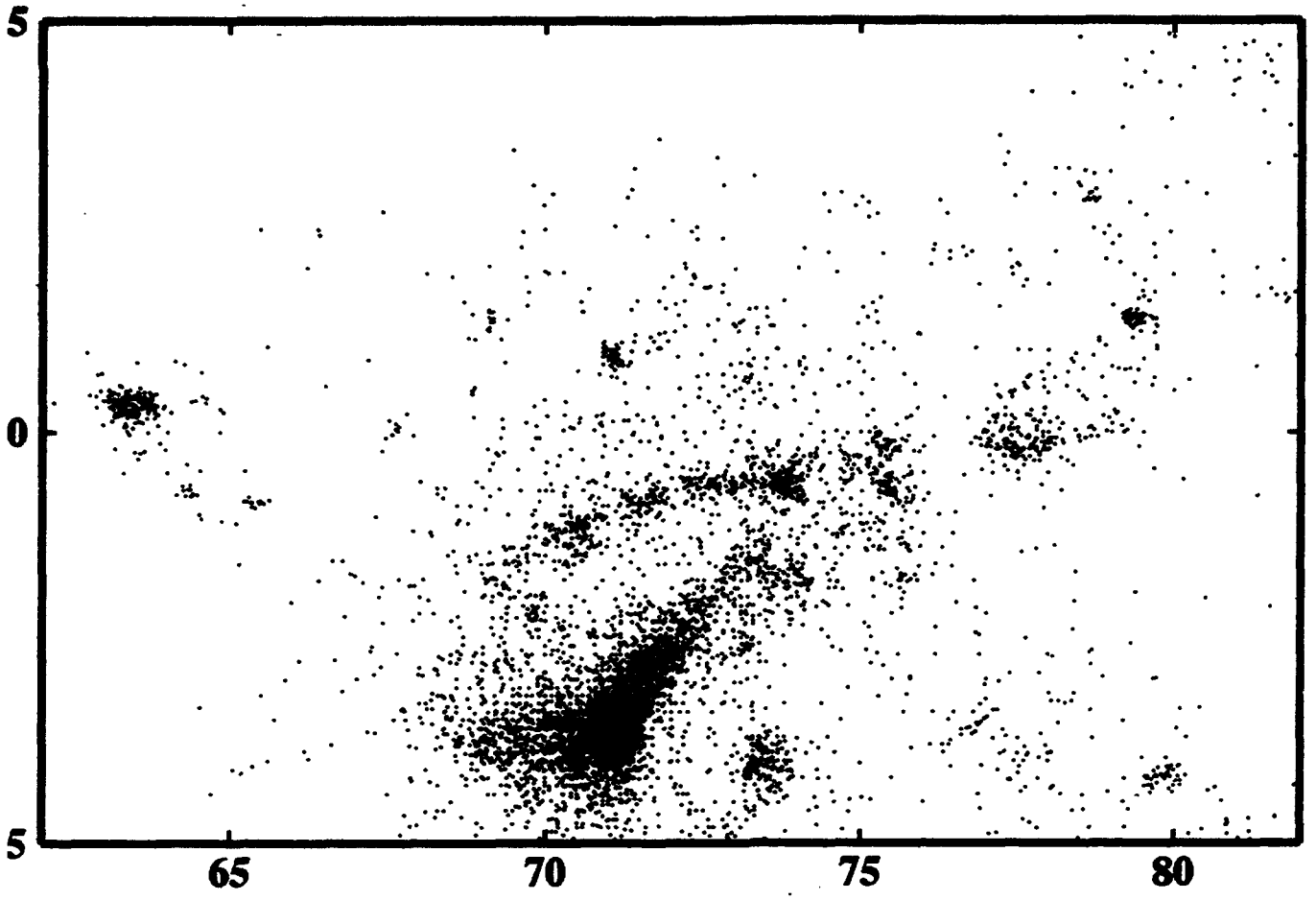
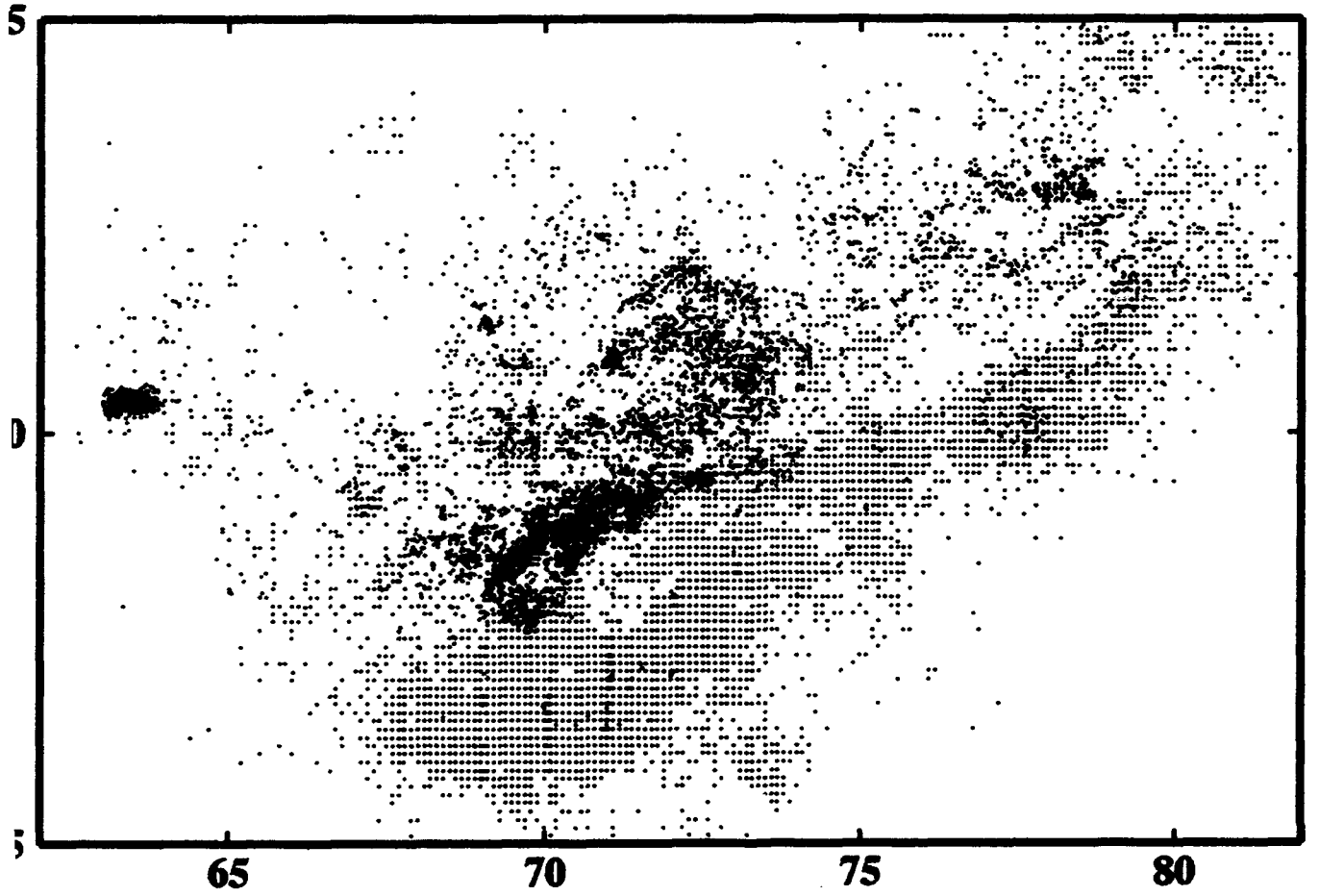


Figure 2: Geography of Central Asia showing political boundaries, main rivers and cities.

Figure 3: Seismicity of Central Asia, 1962 - 1985 from catalogs of the International Seismological Center (ISC, 7808 events, top) and Soviet Academy of Sciences (ZSSSR, 43,425 events, bottom). Each event is shown as a single dot. All depths are included. In the ISC data, the most obvious features are the events of the 1976-78 Gazli sequence in the far west; the E-W Gissar-Kokshal fault zone in the central part of the map and the high activity of the intermediate depth seismicity of the Hindu Kush in the south.


5
0
5
65
70
75
80


5
0
5
65
70
75
80

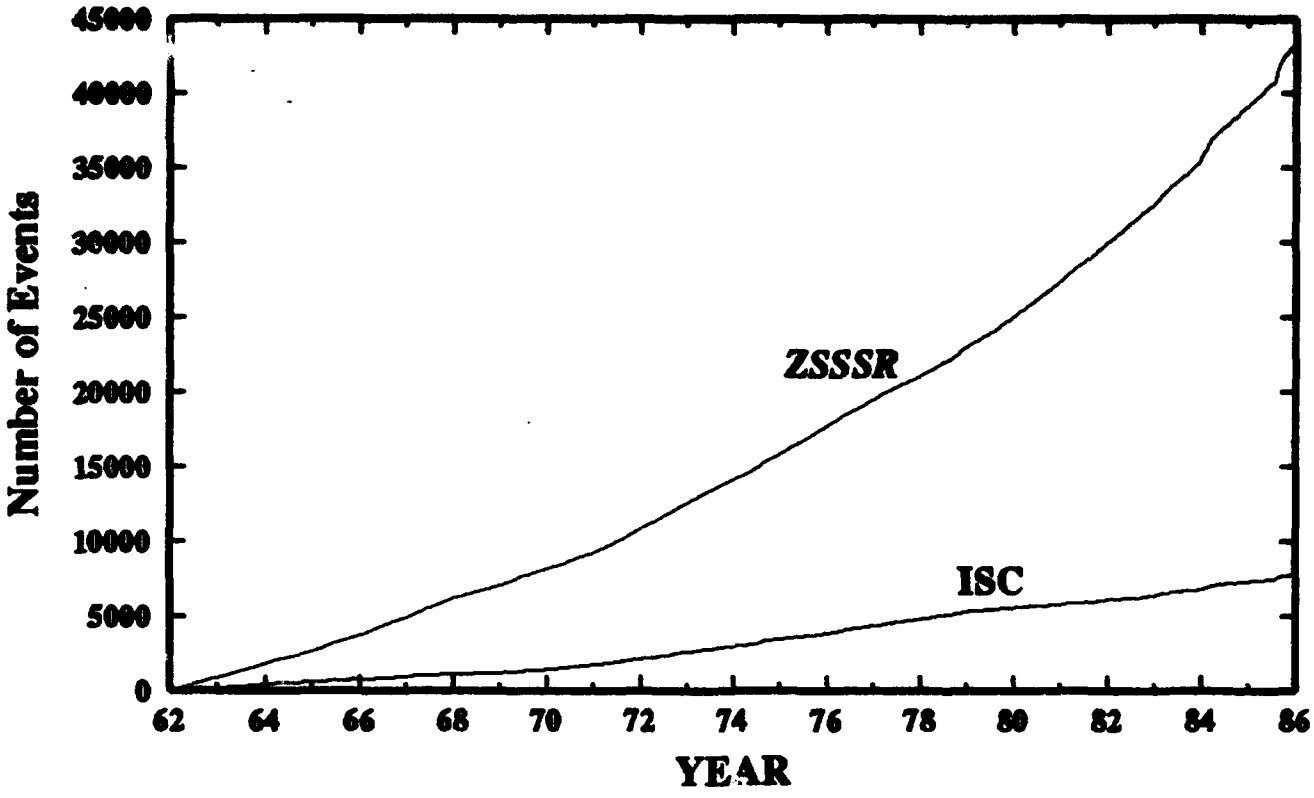


Figure 4: Cumulative number of events in the ZSSSR and ISC catalogs, 1962 - 1986. All events are included, without a lower magnitude cutoff. The increasing slope of the curves, especially for the ZSSSR catalog, is primarily caused by the increasing sensitivity of the catalog to smaller earthquakes.

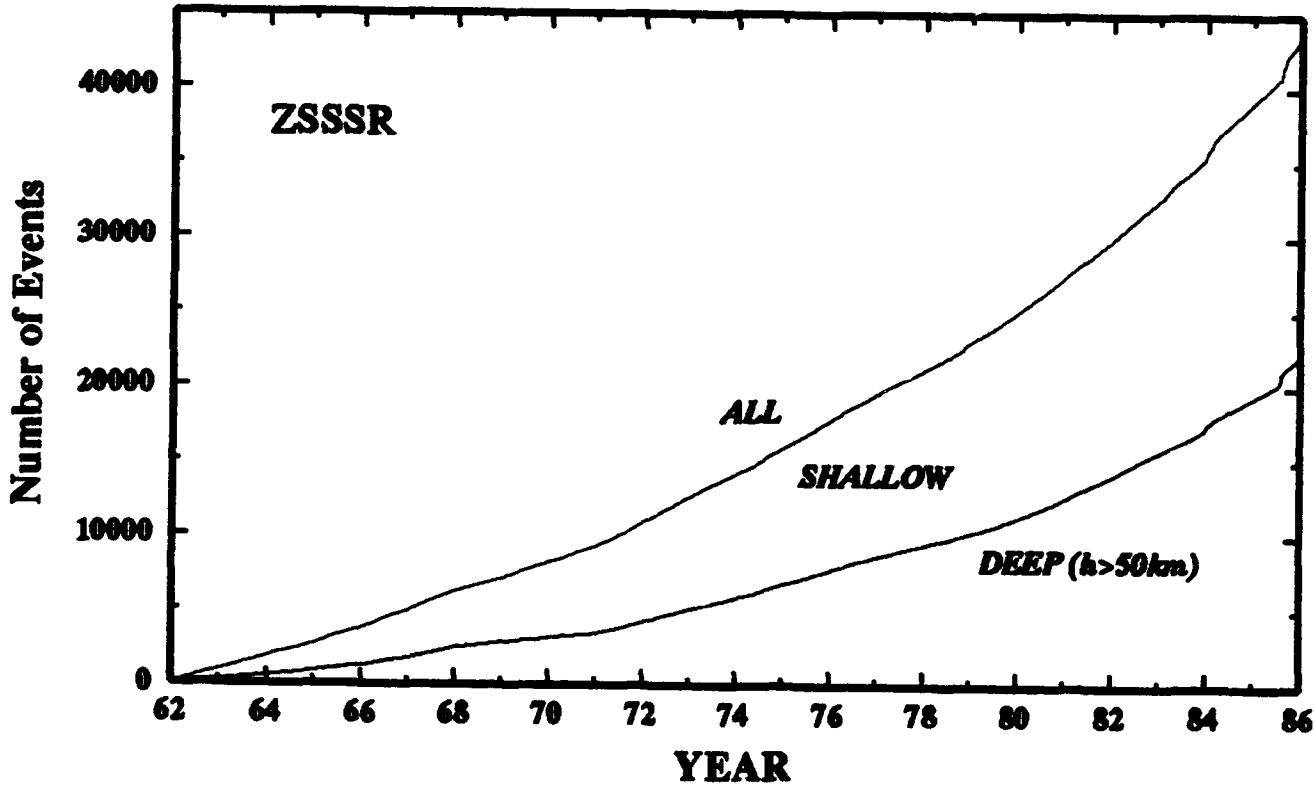


Figure 5: Cumulative number of events in the ZSSSR catalog for all, shallow (<50 km) and deep (>50 km) events. It can be seen that the intermediate depth earthquakes in the Hindu Kush - Pamir zone accounts for almost half of the activity for this region.

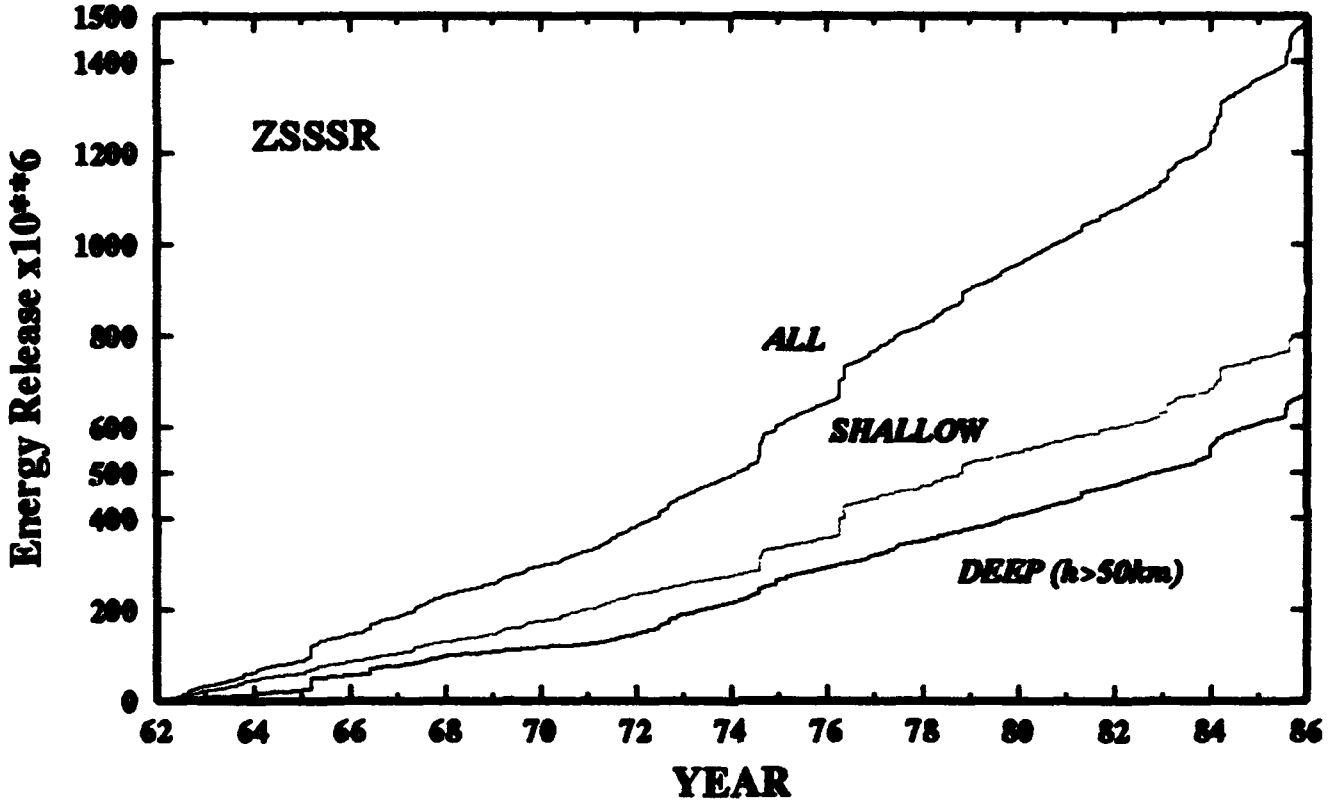


Figure 6: Cumulative energy release from events in the ZSSSR catalog for all, shallow (<50 km) and deep (>50 km) events. The energy is calculated from the size specification for each event in the Soviet catalog which is given in terms of energy class (k), defined as the log of the energy release in joules. An approximate relationship between magnitude and energy class is given by m=(k-4)/1.8. An apparent change in slope in 1972, especially for the deep events, suggests that there was a systematic change in the methods used to determine event size. The larger earthquakes can be easily identified as steps in the energy release plots. Some of the more significant events (greater than magnitude 6.5) include: intermediate depth Hindu Kush in 1965, 1966, 1983 and 1985; Gazli in 1976 (2 events) and 1984; Markansu (1974) and Za-alai (1978) northeast and north of the Pamir and Kokshaal northeast of the Pamir in 1985 and 1983.

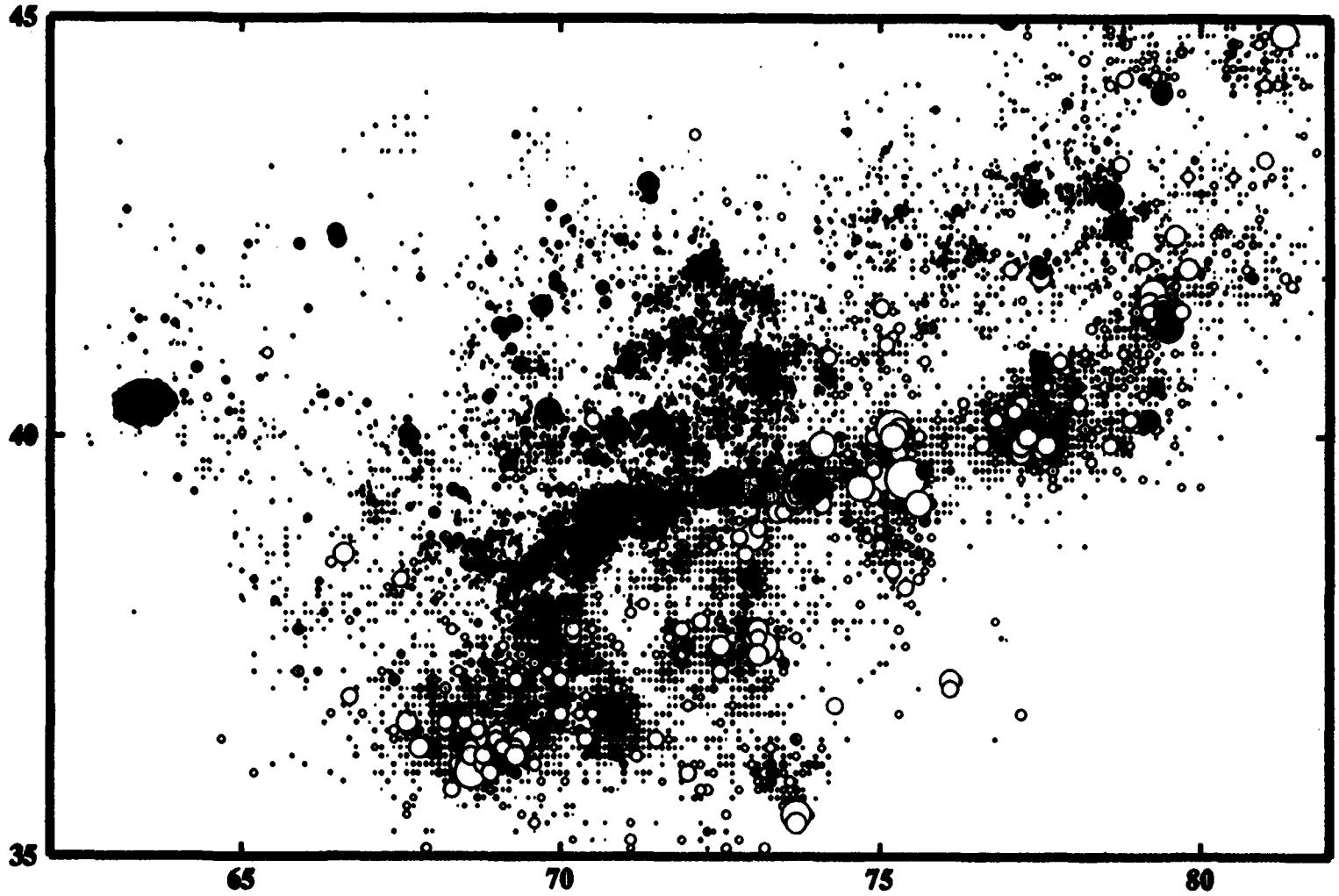


Figure 7: "Shallow" seismicity from the ZSSSR catalog. All events with focal depths specified in the catalog of less than 50 km are shown in red. Events in beige are those for which no depth is specified. Symbol size is proportional to magnitude. It is likely that many of the events in the southern part of the map are deep events of the Hindu Kush.

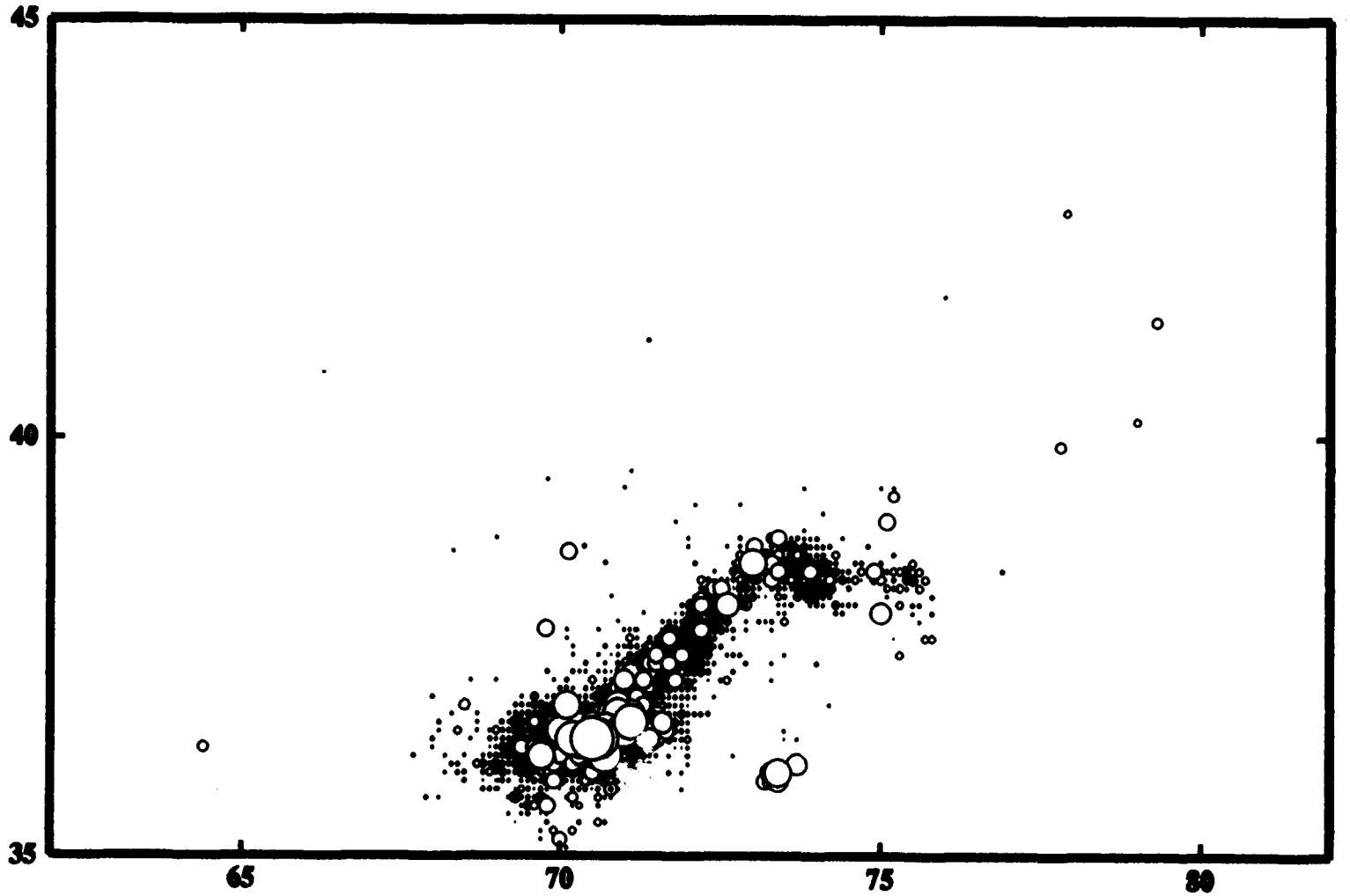


Figure 8: Deep seismicity of Central Asia. All events with focal depths greater than 50 km in the ZSSSR catalog are included. The most obvious feature is the contorted zone of intermediate depth activity beneath the Hindu Kush and Pamir. The highest levels of activity are in the south, where the seismicity dips to the north beneath the Hindu Kush mountains. In the eastern section, the zone dips to the south beneath the Pamir

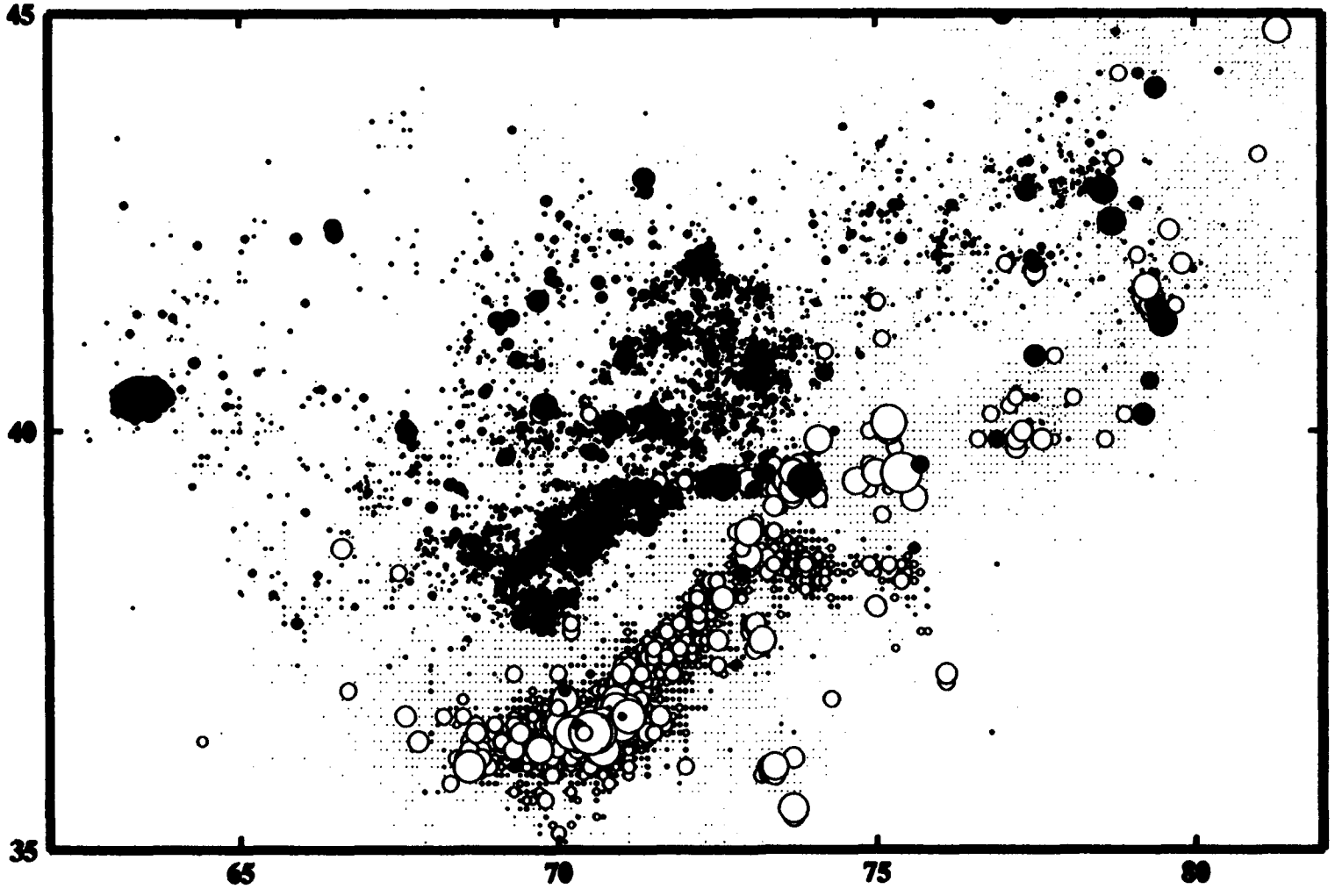


Figure 9: Seismicity of Central Asia from the ZSSSR catalog 1962 - 1986, including both crustal and intermediate depth earthquakes. In this map, those events that are clearly specified in the catalog as shallow are shown as red circles. Shallow events as white circles are those events with energy class greater than 13 (approximately magnitude 5) for which no depth or zero depth is given in the ZSSSR catalog. It can be assumed that these are also crustal earthquakes, since events as large as this would be recorded at sufficient stations to provide a good indication of depth if they were sub-crustal. Events smaller than energy class 13 for which no depth is given are shown as (almost invisible) dots. Events deeper than 50 km are shown in yellow circles.

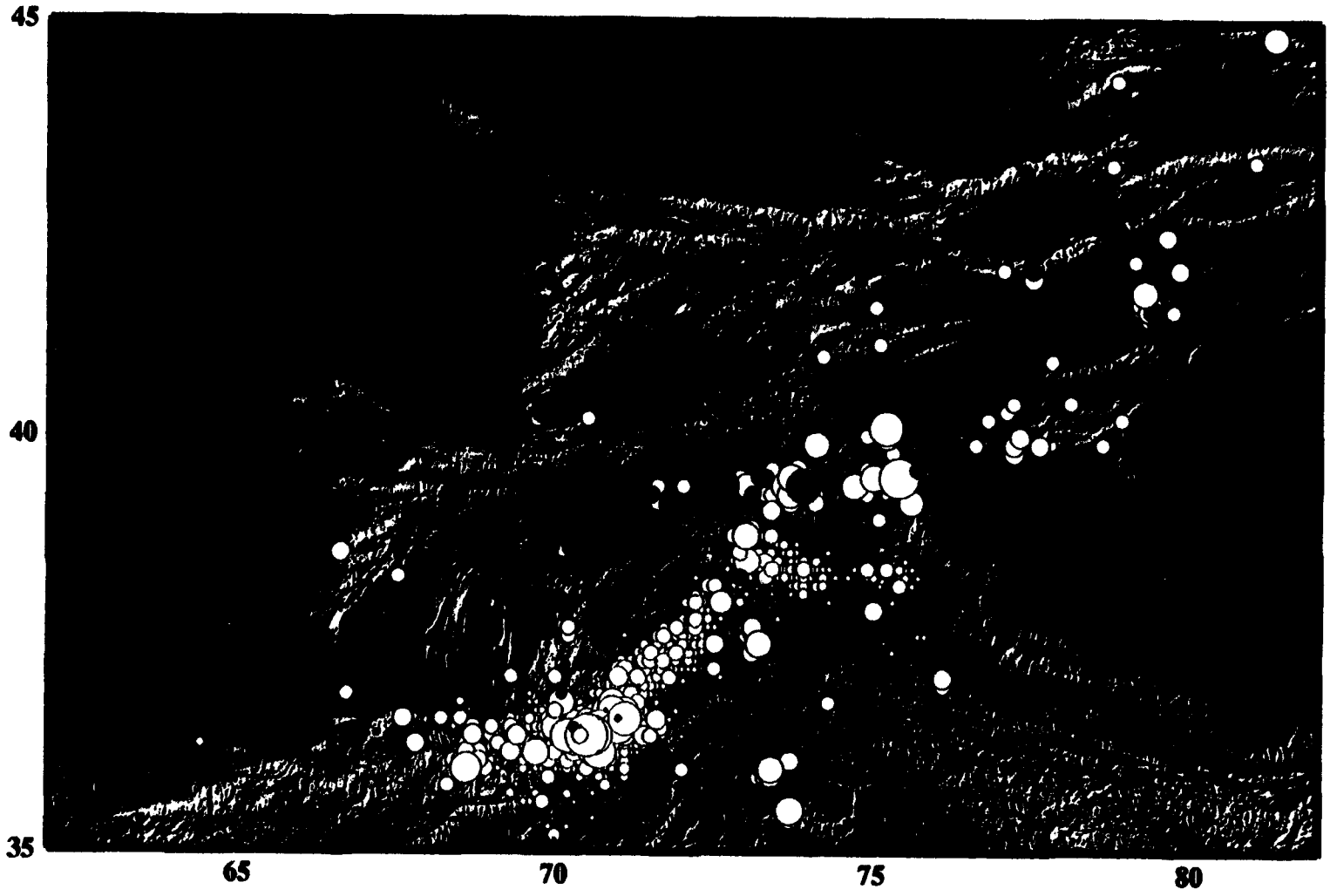


Figure 10: Seismicity map of Figure 9 superimposed on the topographic base map.

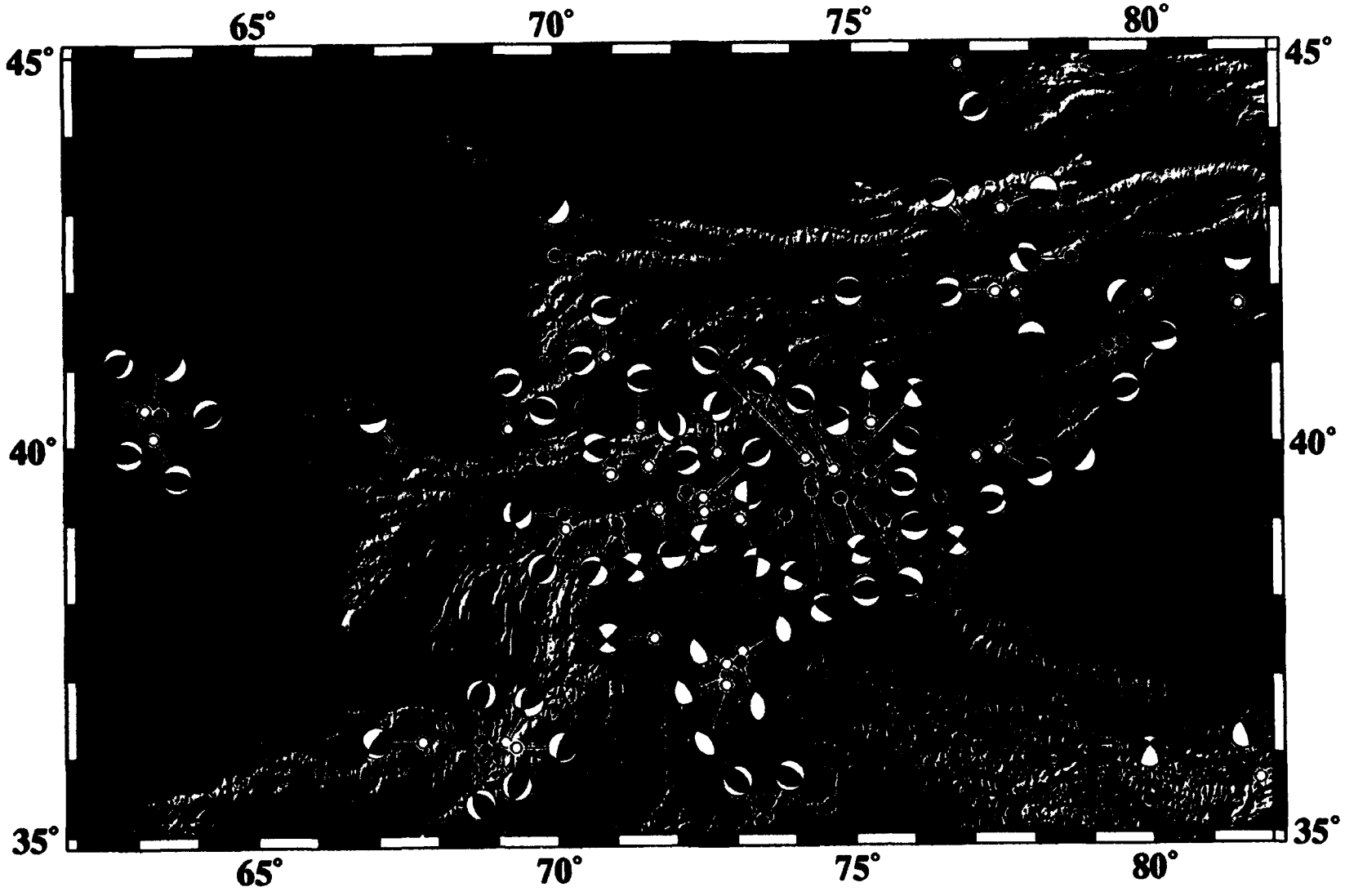


Figure 11: CMT solutions from the Harvard catalog for shallow (crustal) earthquakes in Central Asia. Most mechanisms indicate thrusting consistent with a northwest compression direction. Some strike-slip mechanisms are observed on faults bounding the Pamir. The most anomalous events are the series of shallow normal faulting events

Topography and Faulting in the Western U.S.– An example of digital map making

by David Simpson and Mark Anders

©Reprint from Lamont-Doherty Geological Observatory 1990 & 1991 Report
Lamont-Doherty Geological Observatory of Columbia University
Palisades, NY 10964 (914)359-2900

Topography and Faulting in the Western U.S.– An example of digital map making

by David Simpson and Mark Anders

■ *The ancient art of map making is being revolutionized by the increasing variety and quality of digital data describing the Earth's surface, as well as the availability of computer software and hardware capable of handling the resulting large volumes of high-resolution data. The traditional components of maps — topography, outlines of lakes and rivers, locations of cities, roads and other cultural features — are becoming increasingly available in digital form. Various geological and geophysical data are also being converted to digital form from existing maps or collected initially by digital means. The advent of special Earth-observing satellites, such as Landsat and SPOT, weather satellites and special missions on the Space Shuttle are providing new synoptic views of the Earth. While these new data are providing information that is of fundamental value in isolation, a more significant impact lies in the capability that digital data provides for easy manipulation of the data and for the combination of data from diverse sources in the creation of new maps.*

By its very nature, much of the research at Lamont-Doherty is directly tied to the making of maps. The maps of the ocean floor by Bruce Heezen and Marie Tharp had a major impact on the early development of plate tectonic theory. Past issues of the Lamont Yearbook have highlighted the work of various Lamont researchers in this area (for example, Bill Haxby's satellite altimetry maps and the work of Bill Ryan and others in high-resolution mapping of the seafloor).

The most fundamental expression of the tectonic processes that have shaped the earth is found in the topography of its surface. Topographic data are rich in geological information over a broad range of spatial scales, from the most obvious global-scale divisions between continents and oceans, mountains and plains, to subtle offsets in topography reflecting fault motion and erosion. Major geological provinces typically show distinctive morphological character, reflecting differences in age, rock type, tectonics and erosional history.

The amount of detail that a map can portray depends on the density of the measurements available. At Lamont we now have available topographic data for the entire Earth, both land and seafloor, with elevation values on a grid with spacing between points of approximately 10 km (every 5 minutes of latitude and longitude). These are the data that were used by Margo Edwards to produce a shaded relief map of the world. For some parts of the globe, more detailed topographic data are becoming available for large areas, with values of elevation every 100 m. For the U.S., the most detailed digital topographic data readily avail-

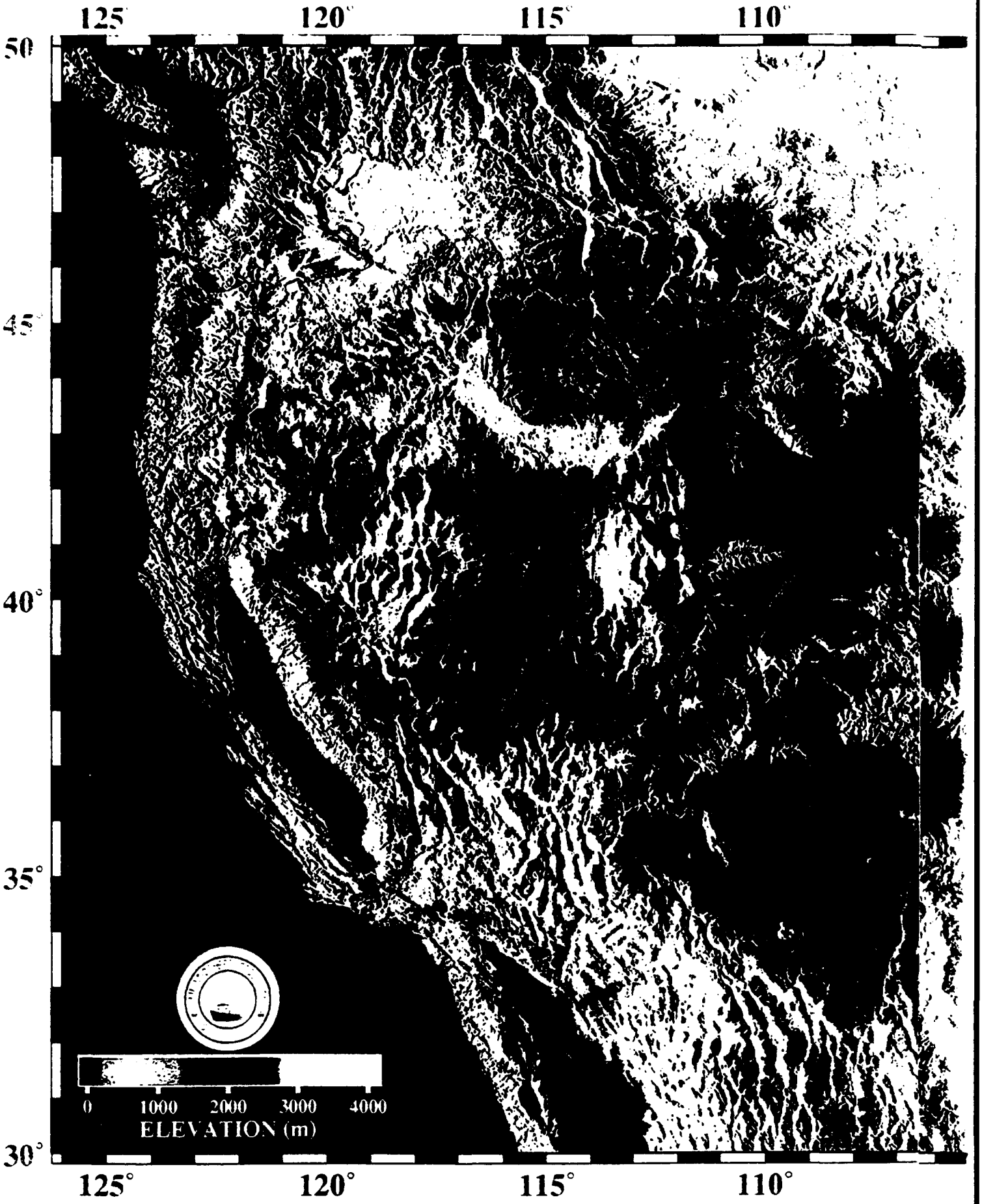

125°
120°
115°
110°
50°
40°
35°
30°
0
1000
2000
3000
4000
ELEVATION (m)

Fig. 1: Topographic map of the western U.S. Shading mimics illumination from the west.

able have been deriv
contour information
topographic maps at
1:250,000. We use h
of this data set that h
between points of 1
term project is now
the U.S.G.S. Nation
graphic Information
will eventually cover
U.S. at point spacing
the order of 10 billio
values). These high-
data will make it pos
tend to quandrangle
ping (on the order of
the types of digital t
that we demonstrate
regional scale.

Physiography of Western U.S.

Physiography is a dir
quence of the intera
tween tectonics and
Both the tectonic an
history can be inferr
relief maps such as F
greater the topograp
the more useful relie
in unraveling the nat
of a particular region
detailed computer-g
lief map of the weste
chose this area to de
of the complex tecto
that has sculptured t
phy, providing a rich
features to interpret.

The tectonic hist
western U.S. is a lon
From the early Paleo
Eocene, the western
derwent a series of c
sional tectonic event
increased its areal ex
thickness. Continent
was increased by the
exotic terrains, and r
al thickening was acc
by emplacement of e

able have been derived from the contour information on existing topographic maps at a scale of 1:250,000. We use here a version of this data set that has a spacing between points of 1 km. A long-term project is now underway by the U.S.G.S. National Cartographic Information Service that will eventually cover the entire U.S. at point spacing of 30 m (on the order of 10 billion elevation values). These high-resolution data will make it possible to extend to quandrangle scale mapping (on the order of 10's of km) the types of digital techinques that we demonstrate here on a regional scale.

Physiography of the Western U.S.

Physiography is a direct consequence of the interaction between tectonics and climate. Both the tectonic and climatic history can be inferred from relief maps such as Fig. 1. The greater the topographic detail, the more useful relief maps are in unraveling the natural history of a particular region. Fig. 1 is a detailed computer-generated relief map of the western U.S. We chose this area to depict because of the complex tectonic history that has sculptured the topography, providing a rich diversity of features to interpret.

: Topographic map of tern U.S. Shading illumination from t.

The tectonic history of the western U.S. is a long one. From the early Paleozoic to the Eocene, the western U.S. underwent a series of compressional tectonic events that both increased its areal extent and thickness. Continental landmass was increased by the addition of exotic terrains, and regional crustal thickening was accomplished by emplacement of eastward di-

rected thrust sheets. Following the cessation of compression in the Eocene, extensional tectonism has dominated in the central and southern portions of the western U.S., and strike-slip tectonism has dominated along southwestern margin of the continent. Throughout the Phanerozoic, volcanism has contributed significant volumes of material to large areas of the western U.S.; this volcanism is both varied in composition and in temporal and spacial patterns of emplacement. The tectonic history of compression, extension, and volcanism has left an indelible mark that can be easily identified by its physiographic expressions.

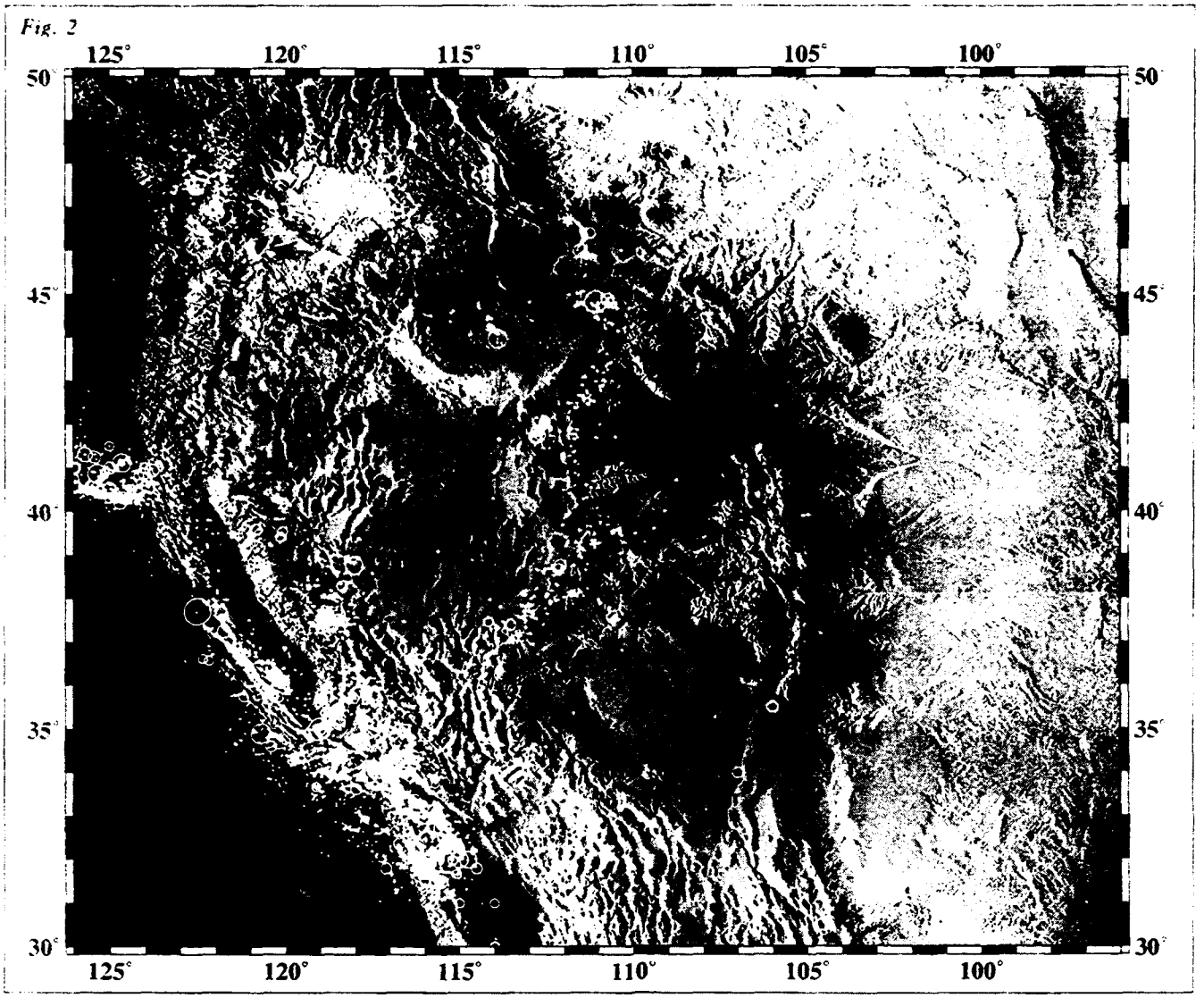
Tectonic history can be interpreted hierarchically, ranging from gross elevation features to those features that are at the limit of resolution on our relief maps. For example, within the western U.S. there is a close correspondence between increased mean elevation and the limits of Mesozoic thrusting. Such a correspondence is best demonstrated in the northeast portion of Fig. 1 by the sharp drop in elevation along the eastern boundary of the Rocky Mountains. Here elevated rocks of the Idaho-Montana thrust belt, as well as the numerous mountain ranges resulting from the "thick-skinned" Laramide thrusts, form a precipitous contact with the unfaulted lower-lying sediments of the western Great Plains. Fine scale observations can also provide important clues to regional tectonic history. For example, note the small "bumps" located on the northeast trending depression that is show in **Figs. 1 and 5**. This depression is the eastern Snake River Plain whose age is less than 15 Ma. On the larger-scale map the bumps are even more evident. These features are volcanoes whose detailed topographic expression suggests that they are both geologically young (< 10,000 yrs.) and similar in age over the entire eastern Snake River Plain. From this it follows that the eastern Snake River Plain is a region that is presently underlain by magma.

Interpretation of Mountain Growth Based on Shape

Although the mean elevation of the western U.S. was most certainly increased during the compression that ended in the early Tertiary, the uplift of many of the higher mountains is much younger. Much of this younger uplift is likely due to an isostatic response to normal faulting, as well as an isostatic response to accelerated erosion rates. Faulting-induced uplift produces the characteristic wedge-shaped profile evidenced in the physiography of a number of mountain ranges in the western U.S. **(Figs. 1 and 4)**. The shape of a mountain range, therefore, can be used to assess the mechanism responsible for its growth.

Surrounding the Great Basin is a belt of seismicity **(Fig. 2)** that also corresponds to some of the region's most active normal faults. Mountain ranges on the eastern margin of the Great Basin exhibit a gentle eastward tapering in elevation and a precipitous fault-bounded escarpment on the western side **(Figs. 4b and c)**. On the western margin, the reverse elevation profile is observable. On the western margin of the Great Basin, the Sierra Nevada Mountains slope gently toward the Great Valley of central California and are steeply truncated on the east by the Owens Valley fault system. Based on the tilting history of young volcanic rocks, the Sierra Nevada Mountains acquired their westward tilt some time after 10 Ma. Similar young uplift histories in association with wedge-shaped tilted mountains are found in the northeastern Basin and Range. For example, prior to 4 Ma the Teton and Centennial Ranges had little or no relief. With the exception of the Uinta Mountains of north-central Utah, most of those high mountain ranges (represented by a white contour interval) with wedge-shaped profiles are bounded by the region's most active normal faults.

Interestingly, some recent publications have suggested that mountain ranges have either grown in response to climate changes (Molnar and England, 1990) or, to the contrary, that increased tectonically driven mountain building has changed climate patterns (Ruddiman and Raymo, 1988). Mountain ranges such as the Uinta Mountains of northern Utah (the east-west trending mountain range that looks like the back of a scorpion) exhibits a symmetric shape suggesting its growth is unrelated to an isostatic response to normal faulting. The Uinta Mountains' shape provides an important clue to its tectonic origin that can be followed up by more detailed geologic studies. Using shape as a tectonic tool, other high mountain ranges in the western U.S. can be examined and assessed as to their possible climatic or tectonic origin.



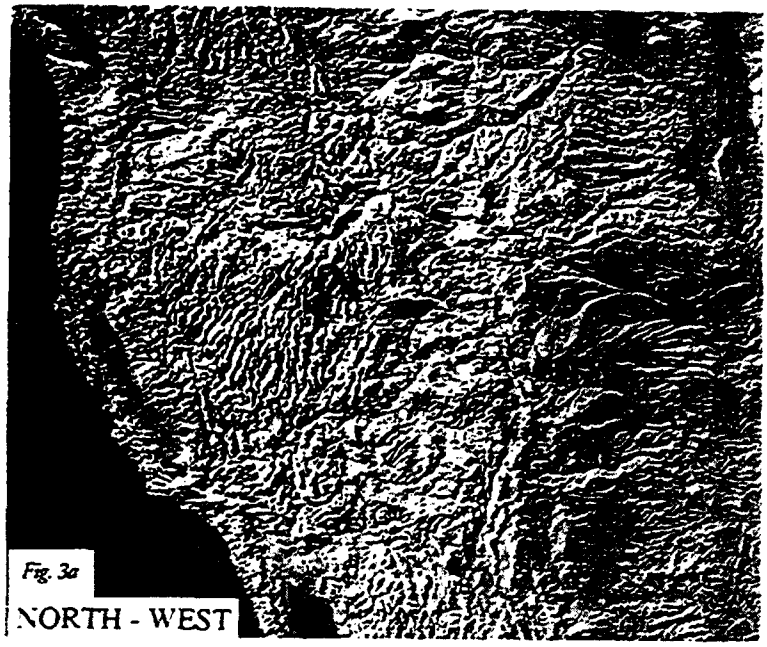
Other Physiographic Features of the Western U.S. and their Tectonic Significance

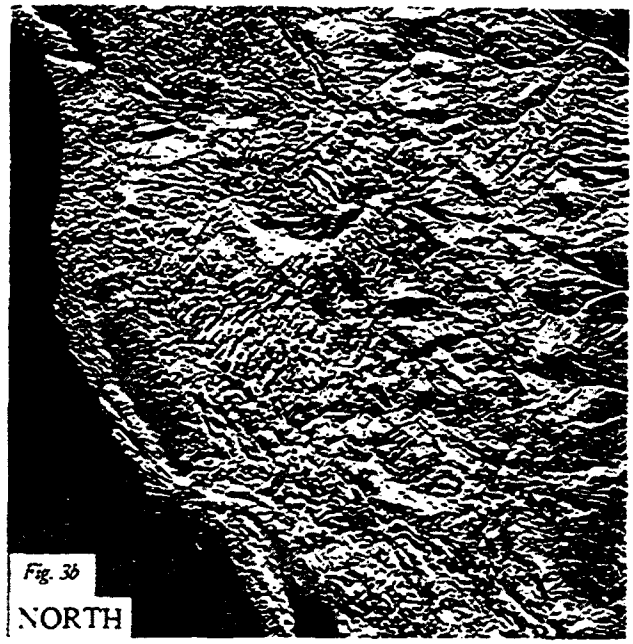
Throughout the western U.S. there are a number of salient physiographic features most of whose origins are well known, though some need further definition. **Fig. 3f** shows the boundaries of many of these physiographic provinces:

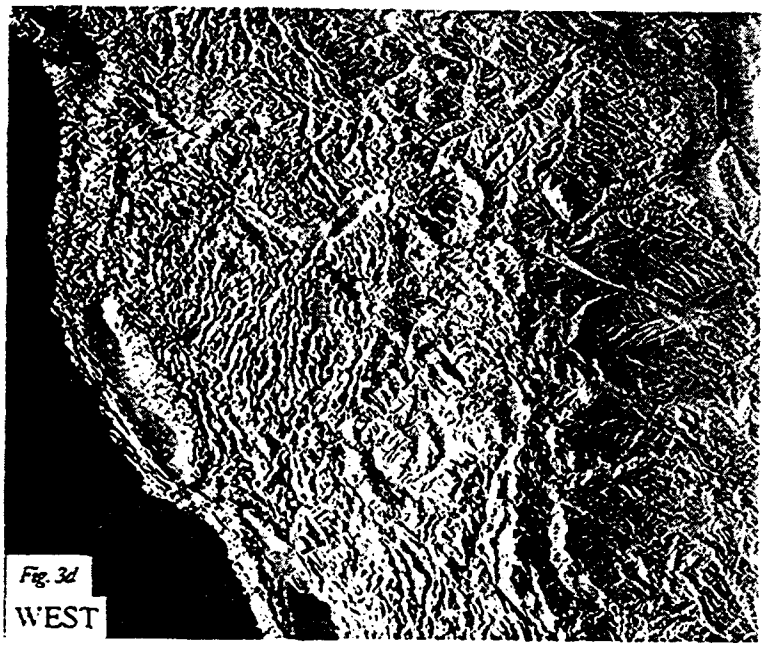
■ *The Basin and Range* is one of the most areally extensive physiographic provinces on our map — and one of the most tectonically complex. Within the Basin and Range there are numerous north-south trending fault blocks that were formed by high-angle normal faults, many of which are presently inactive. Both the active and inactive faults, and associated fault blocks, form a pattern of north-south trending ranges that have been described as "snakes slithering south." This north-south trend reflects a roughly east-west extension direction during the late Cenozoic. This region has variously been described as having extended during the Tertiary anywhere from 50-300%.

■ Located within the bounds of the Basin and Range is the sub-region known as the *Great Basin*. The central portion of the Great Basin is located between two regions of elevation lows. Why these areas are depressed relative to surrounding areas is

Fig. 2: Seismicity of the western US. Open white circles, with size scaled to magnitude, are large earthquakes with magnitude greater than 6, 1900-1975. Yellow dots are earthquakes of magnitude greater than 2.5, 1970-1985. Epicentral data are from the Decade of North American Geology (DNAG) catalog of Engdahl and Rinehart (1989).







not clear, although the apparent uniformity of basin floors may be due in part to sedimentation from lakes that occupied this area during the late Pleistocene.

Several prominent physiographic features observable in Fig. 1 are due to extrusion of large volumes of volcanic rock. They are the north-south trending Cascade Range, the Columbia Plateau and the Snake River Plain. Two of these features form large flat surfaces, and the other, the Cascade Range, forms a mountain range made up of volcanoes, many of which are observable as the chain of cone-shaped peaks trending north-south on the northwestern portion of Fig. 1.

■ The *Cascade Range* was formed by continuous andesitic volcanism from the Eocene to the present. The Cascades extend from Canada to the northern tip of the Great Valley of central California. In Fig. 1 the southernmost volcano of the Cascade Range, Mt. Shasta, is marked by a white elevation contouring.

■ The large, relatively flat area to the east of the Cascade Range is the *Columbia Plateau*. The Columbia Plateau is an ex-

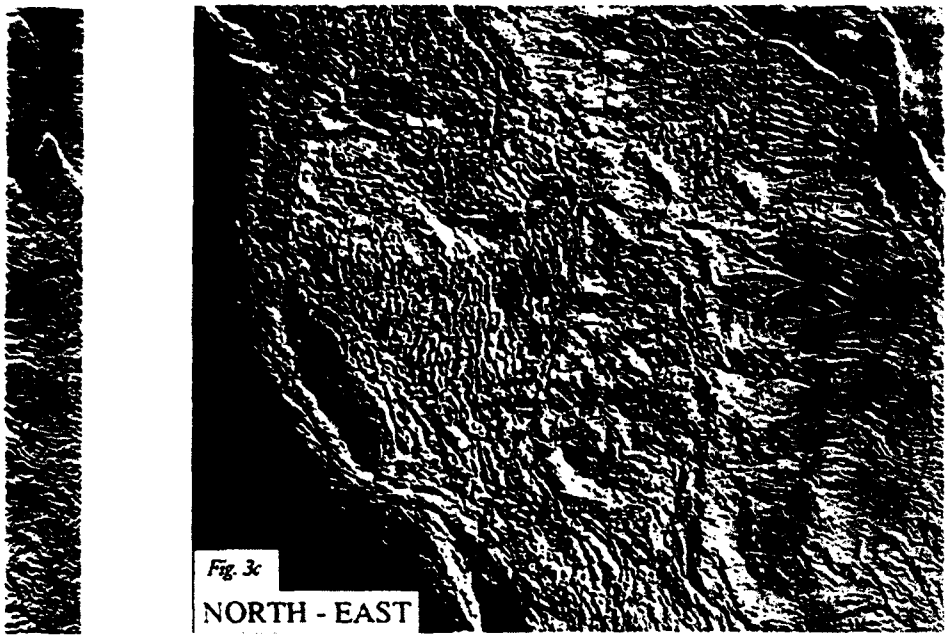
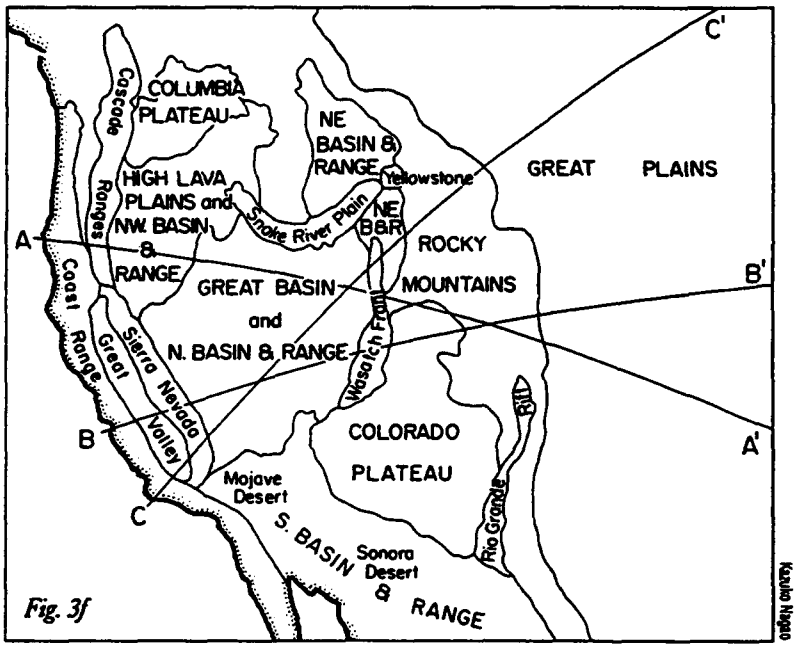


Fig. 3a-d: Shaded relief maps for the western U.S. with illumination from different compass direction. These different directions highlight different features in the topography, with emphasis on those structures perpendicular to the direction of illumination.
3e: Gray-scale map of the topography of the western U.S. Fig. 1 is created by combining a colored version of this map with the gradient map in (d).
3f: Major physiographic features of the western U.S. Lines AA', BB' and CC' indicate the profiles used for the cross sections of topography shown in Fig. 4.

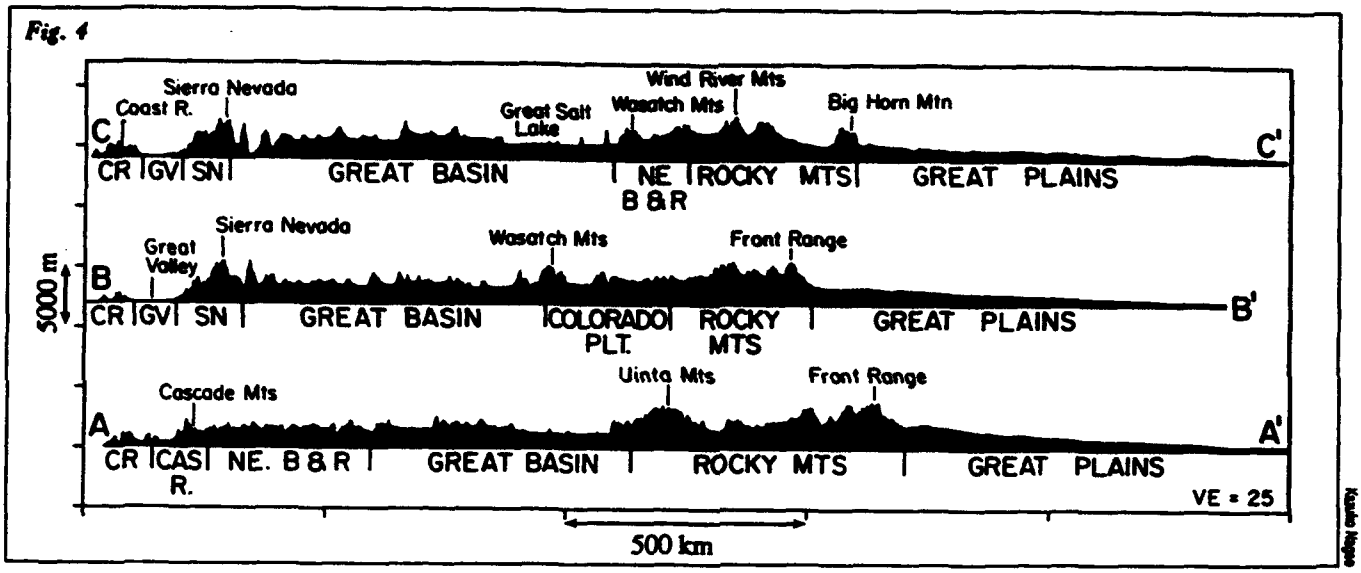


tensive region of tholeiitic flood basalts that were mostly extruded in less than two million years around 17 Ma.

■ To the southeast of the Columbia Plateau is the *Snake River Plain*, which is a prominent semi-circular depression filled with 1 to 3 km of tholeiitic basalts. The Snake River Plain is divided into a western and eastern branch, of which the eastern branch relates to the track of a hotspot, or sub-lithospheric magma plume, that is thought to have initiated the Columbia Plateau flood basalts. There is a progressive decrease in the age from the volcanism that first initiated on the Owyee Plateau (the circular region of raised elevation located between the western depression of the Great Basin and Snake River Plain). Currently, the hotspot is thought to be located somewhere under Yellowstone National Park. The concentration of seismicity seen at the northeastern extent of the eastern Snake River Plain is probably due to magmas associated with the present position of the hotspot.

■ The greatest concentration of seismicity in the western U.S. is associated with the San Andreas fault along the southwestern margin of the continent. Physiographically, there is a clear pattern of west- to northwest-trending mountains that comprise the California *Coast Ranges*. The trend of the California Coast Ranges reflects the former and present trend of the San Andreas, as well as numerous subsidiary strike-slip faults. South of the Great Valley the strike-slip fault system directly abuts the Basin and Range province.

■ To the east of the Coast Ranges Jurassic to Cretaceous sedimentary rocks of the *Great Valley* are folded into a large synclinal structure that is in turn overlain by Cenozoic marine and terrestrial deposits. A history of marine transgressions and lake



impoundment are represented by the flat lying sediments that comprise the floor of the Great Valley.

■ A large and uniformly elevated surface in the south central portion of our relief map is the *Colorado Plateau*. In Fig. 1 the Colorado Plateau is seen as a large brown mass bounded on the west, northwest and southwest by the ridge and valley system of the Basin and Range province. The Colorado Plateau is a large contiguous uplifted block of Paleozoic/Mesozoic sediments lying on a Precambrian basement, all of which have been undeformed by either compressional or extensional tectonism.

■ The final physiographic feature to be discussed is the most spectacular. The central *Rocky Mountains* have the highest mean elevation of any of the provinces discussed above. There are well over 50 mountains that have elevations of over 14,000 ft. in this region. The central Rocky Mountains are part of the world's only foreland, or interior, orogeny. Much of the central Rocky Mountains developed by thrusting on high-angle (~30° dipping) thrust surfaces. Their origin may be related to traction between the North American Plate and part of the subducted Pacific Plate during the 80 to 40 Ma Laramide orogeny. Trending straight south from the central Rocky Mountains is a depression that represents the currently extending Rio Grande Rift. The development of the Rio Grande rift thus completes the regional pattern of thickening of the lithosphere of the western U.S. followed by a subsequent reduction in mean elevation by extensional tectonism.

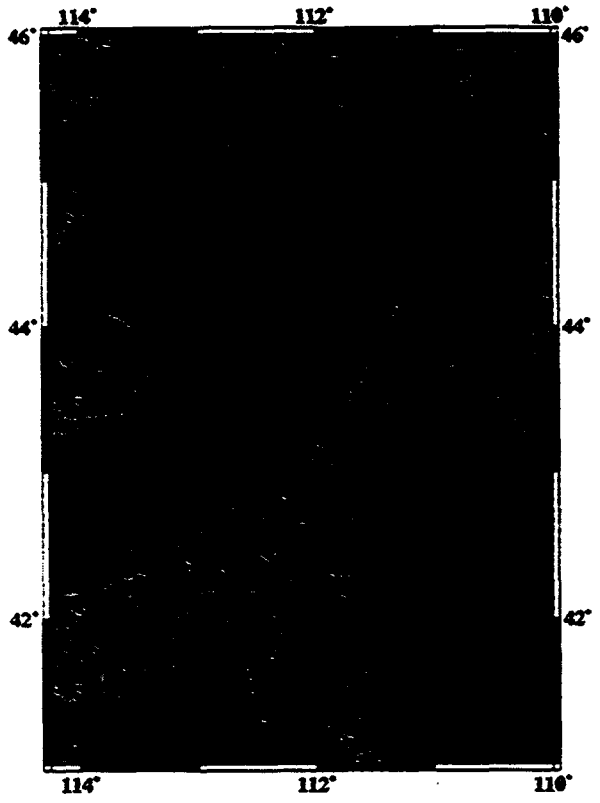
The value of our computer-generated relief map lies in its ability to resolve physiographic details that are otherwise only available by patching together Landsat images or by combining large numbers of detailed topographic maps. The chief advantage over Landsat is the capability for comparing relative elevation patterns, as well as for overlaying information such as earthquake and fault data so that it can be analyzed on a regional basis. The advantage over topographic maps lies in the detailed imagery previously only obtainable by viewing one map at a time. Our topographic relief maps may offer their greatest advantage in such intangible ways as allowing tectonists to view the "Big Picture" while maintaining the "detail" that is a necessary quality of any tectonic synthesis.

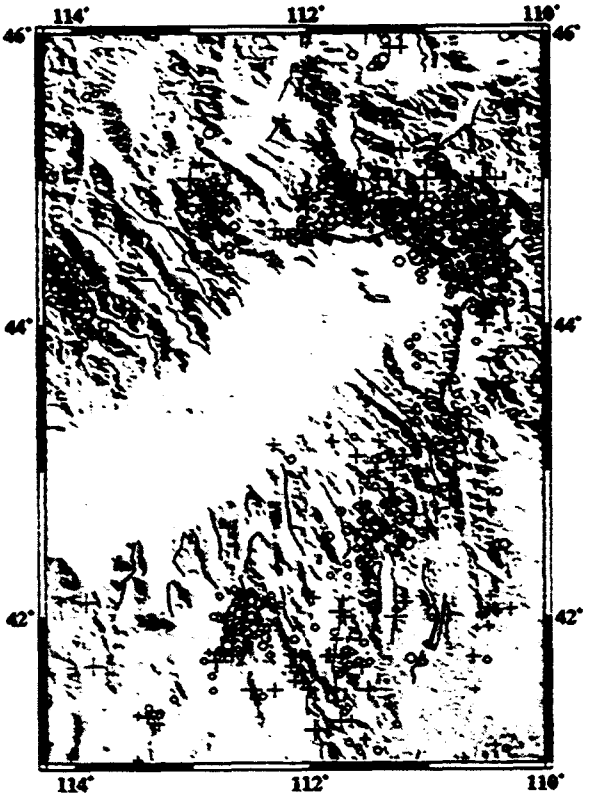
Fig. 4: Topographic profiles of the western U.S. shown in Fig. 3f. Major physiographic provinces are indicated. Profiles are plotted with a vertical exaggeration of 25.

Seismicity and Late Cenozoic Faulting of the Northeastern Basin and Range Province

As an example of a more detailed analyis of a smaller geographic area, Fig. 5 shows the topography and seismicity of the northeastern Basin and Range. This region experienced significant northwestward-directed thrusting from the Jurassic to the early Eocene during what is called the Sevier orogenic event. The di-

Fig. 5a





rection of thrusting is reflected in the distribution of parallel arc-shaped mountain ranges in this region (Fig. 5a). Subsequent to the Sevier orogeny the northeastern Basin and Range has undergone two major extensional events. The first was an east-west directed extension in the late Eocene that appears to be short-lived, lasting only a few millions of years. The second event was a northeast-southwest- to east-west directed extension that started sometime in the Miocene (exactly when is not clear) and has continued to the present. The second event is observable by the deep extensional valleys positioned in an arcuate fashion between parallel formed mountains.

The most dominant physiographic feature in the northeast Basin and Range is the Snake River Plain. The Snake River Plain is a depression filled with several km of rhyolitic volcanic rocks and sediments capped by 1-3 km of basalt. Intruded within the crust are another 5 to 10 km of basaltic material. The sequence of rhyolitic volcanism followed by basaltic volcanism has migrated to the northeast across the entire length of the eastern Snake River Plain during the last 17 Ma. Note again, the Holocene basaltic volcanic centers that were discussed above are

Fig. 5a: Topographic map of the Northeastern Basin and Range.

5b: Seismicity and faulting in the Northeastern Basin and Range, superimposed on a map of the topographic gradient. Earthquake epicenters for historical earthquakes are shown as crosses, recent earthquakes as circles. Earthquake data are from the DNAG compilation, with most of the recent seismicity in this area reported by the University of Utah seismographic network. Faults are color-coded by age with red for Historical ruptures, purple for latest Quaternary, light blue for Quaternary and green for late Cenozoic.

Making the Maps

The topographic data used to produce the maps of the western U.S. shown here were originally derived by the Defense Mapping Agency from 1:250,000-scale topographic maps. The elevation contours on these maps were scanned electronically and a raster (equally-spaced grid of data points) was produced by interpolating between contour lines. These data, along with a variety of other geophysical data sets (gravity, magnetics, seismicity), have been distributed on an optical disk entitled "Geophysics of North America" by the National Oceanic and Atmospheric Administration (NOAA).

Our work on the map began with a computer file containing an evenly spaced grid of numbers representing the elevations of points on the Earth's surface. The grid spacing is 30 arc seconds in both latitude and longitude (120 points per degree, or approximately 1 point per km). For the part of the western U.S. used here, this represents almost 8 million individual elevation values.

When projected onto a map coordinate system with each point ascribed a brightness proportional to height above sea level, the resulting map shows high elevations as white and low areas as black (Fig. 3e). This representation retains the information on absolute elevation and accentuates the broad scale changes in topography and texture; for example the broad, flat and relatively low elevations of the Great Valley of California, the Snake River Plain and the Colorado plateau contrasted with the high and rugged topography of the Rockies or the Sierra Nevada.

Shorter-scale changes in elevation are more clearly seen when, instead of the absolute elevation, we display the change in elevation at each point. This change in elevation, or slope, is calculated as the difference in elevation between adjacent points on the elevation grid. If the brightness on the map presentation is proportional to the slope, flat areas appear as neutral grey, strong positive gradients, or uphill slopes, appear bright, while strong negative, or downhill, slopes appear dark. Since this mimics the effect of the illumination of a three dimensional object from the side (bright slopes facing the light, with shadows on the far side) the resulting maps have the appearance of an overhead view of the land surface lit by the sun low in the horizon.

Since the slope at a point can change, the slope map will have a different appearance depending on the direction in which the slope measurement is made. Consider, for example, a long linear ridge. It is clear that walking along the flanks of a ridge the slope is less than walking directly uphill; if the gradient is measured in a direction parallel to the ridge, there is little change in slope and the appearance of the feature is suppressed.

If the gradient is measured perpendicular to the ridge, there is a rapid change in slope and the ridge is enhanced, with a bright side in the direction of the gradient (or illumination) and a shadow on the far side. This is seen in Fig. 3d where the NS structures in the Basin and Range are most prominent when the gradient is measured from the west and subdued when illuminated from the north. An even clearer difference is observed for the San Andreas system, where the predominant linear trend to the NW is clearly enhanced with illumination from the NE, but almost disappears with illumination from the NW.

In the color map (Fig. 1), the height and gradient information are combined by using color to represent absolute elevation and brightness to represent the gradient. A color scheme was devised to ascribe smoothly varying colors to increasing elevation, varying from blue at sea level, through green to orange and finally white for the highest elevations. The color value for each point was then multiplied by a value corresponding to the gradient at that point. Since the gradient map produces an effect similar to illuminating a three dimensional surface with the light source in the direction of the gradient (bright on slopes that face towards the light; dark on the far side) the final map has a 3-D appearance, in this case as if illuminated by a sun setting in the west.

The earthquake data used in Figs. 2 and 5b are from the compilation of Engdahl et al. (1989) prepared for the Geological Society of America's Decade of North American Geology. For this compilation, epicenters from a variety of historical and instrumental catalogs were combined to provide relatively uniform coverage over the United States. Larger earthquakes from historical and recent catalogs indicate the main areas of major seismicity, while lower magnitude seismicity, from regional networks operating over the past 15 years, show more clearly the details of spatial trends in activity. Shown in Fig. 2 as white open circles are 118 earthquakes of greater than magnitude 6.0 from 1900 to 1975 and, as yellow dots, 18077 earthquakes greater than magnitude 2.5 from 1970 to 1985.

visible as small "bumps" on the eastern Snake River Plain. The pattern of seismicity (epicenters represented as yellow circles in Fig. 5b) forms a symmetric shape centered around the track of the migrating Yellowstone hotspot. This shape has variously been defined as an "arc," a "V" and a "parabola." Although surrounded by intense seismic activity, the eastern Snake River Plain itself is a region of both aseismicity and high heat flow. In 1983 it was first suggested that there is a regional pattern of increased rate of faulting that was followed by a cessation of faulting which correlated directly with the migration of the Yellowstone hotspot.

Within the northeastern Basin and Range shown in Fig. 5b there are a number of active high-angle normal faults that have been color-coded according to the time of their last movement. Red is for historic ruptures, purple is for latest Quaternary (<15,000 yrs.), light blue is Quaternary and green is for late Cenozoic (roughly < late Miocene). It can be clearly seen that the distribution of historic and latest Quaternary faulting mimics the symmetric pattern of seismicity about the eastern Snake River Plain. Furthermore, there is a close correlation between high peaks (white contours) and historic-to-latest Quaternary faulting. Combining all the above information onto one diagram greatly enhances our ability to interpret regional patterns of tectonism.

References

The earthquake data used in Figures 2 and 5 are from the compilation of Engdahl et al. (1989) prepared for the Geological Society of America's Decade of North American Geology. For this compilation, epicenters from a variety of historical and instrumental catalogs were combined to provide relatively uniform coverage over the United States. Larger earthquakes from historical and recent catalogs indicate the main areas of major seismicity while lower magnitude seismicity, from regional networks operating over the past 15 years, shows more clearly the details of spatial trends in activity. Shown in Figure 2 as white open circles are 118 earthquakes of greater than magnitude 6.0 from 1900 to 1975 and, as yellow dots, 18077 earthquakes greater than magnitude 2.5 from 1970 to 1985.

Molnar, P. (M.I.T.), and England, P. (Oxford), 1990, Late Cenozoic uplift of mountain ranges and global climate change: chicken or egg?: Nature, v. 346, n. 6279, p. 29-34.

Ruddiman, W.F., and Raymo, M.E. (Lamont), 1988, Northern hemisphere climate regimes during the past 3 Ma: possible tectonic connections: Philosophical Transactions of the Royal Society of London, v. B318, p. 411-430.

Engdahl, E.R., Rinehart W.A., 1989, Seismicity Map of North America in Slemmons, D.B, Engdahl, E.R., Blackwell, D. and Schwartz, D., Neotectonics of North America. *Boulder Colorado, The Geological Society of America, CSMV-1.*

Image Processing

in Seismology

by Irit Dolev

1.0 Preface

This document is a general outline of the image processing software available in Seismology. The document mentions only the main relevant programs. Sometimes, details are sacrificed for the sake of clarity. The reader is referred, in the appropriate sections of this document, to further reading of manuals.

The diagrams are made using the following signs:
An oval is a process, software, action.
Two parallel lines contain a file, data.
An arrow shows the direction of data flow.
A rectangular is a HW device: workstation, printer, tape drive, M75 system.

Processed tapes are on the rack near the Iris printer.
The list of tapes can be found in ~irit/tapes/tartapes/tapelist.
Processed files are saved on tapes by using the tar command.

2.0 Overview

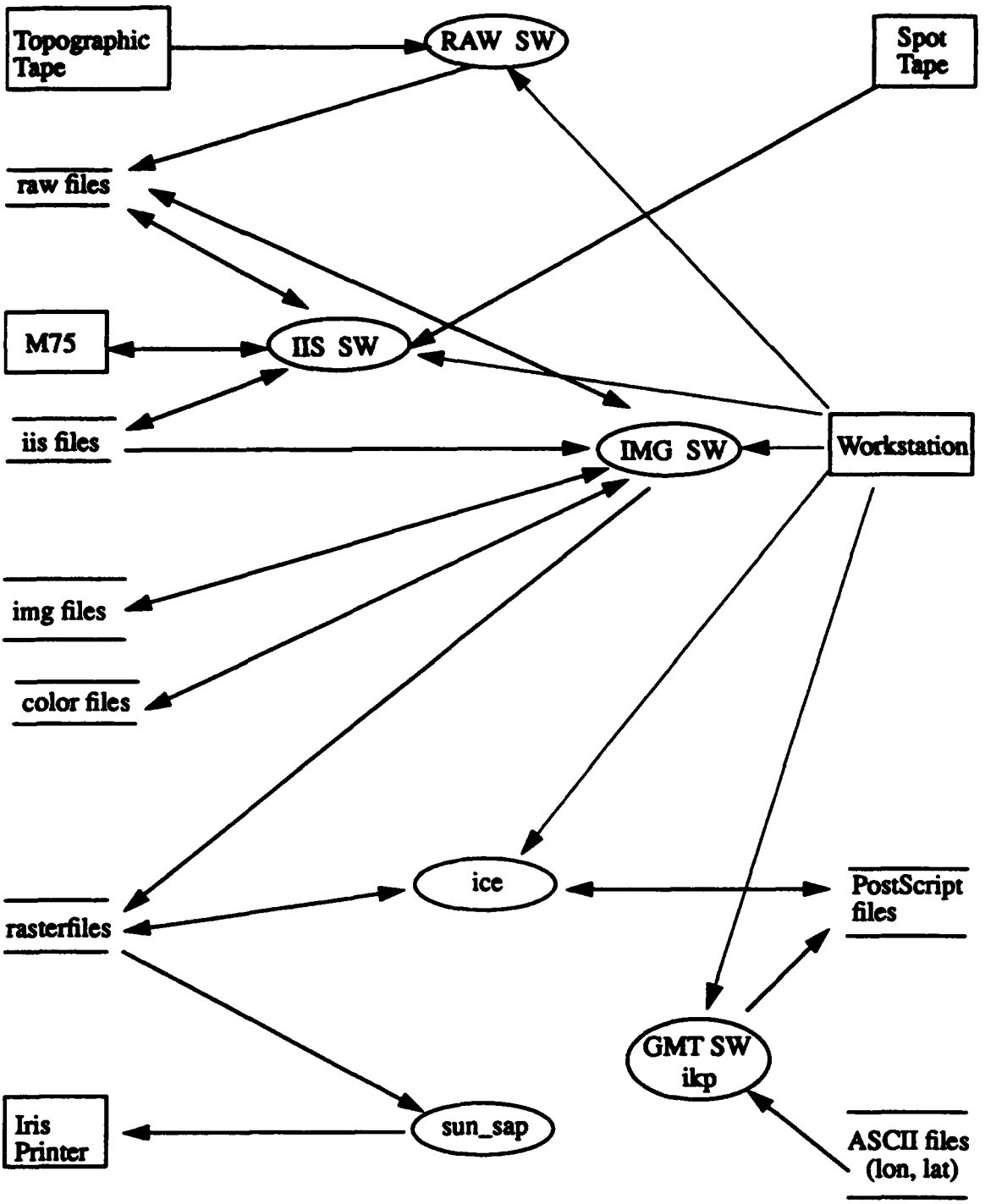
Look at the diagram on page 2 (Overview) for reference. Chapters 3 and 4 describe the software (SW) and data formats that are mentioned in the diagram.

Raster data for images come from tapes, cd-roms, etc.
Topographic tapes are read and then merged with the RAW SW. The output from these programs are raw files. Raw files are converted to iis files (with the IIS SW) or to img files (with the IMG SW).
Spot images are read with the IIS SW which converts them to iis files.

Image processing is done with the IIS SW (on iis files) or with the IMG SW (on img files). The final processed files are converted to rasterfiles that are printed with the sun_sap command on the Iris printer.

Sometimes it is desired to add annotations (symbols, lines, etc.) to the rasterfiles. The GMT SW is used to generate annotations in Postscript format. The ikp program is used as an aid in the construction of GMT processing pipelines. The rasterfiles and PostScript files are then combined with the ice program to create a final rasterfile. This file is printed on the Iris printer.

FIGURE 1. Overview



3.0 Software

3.1 RAW SW

Programs to read topographic data from tapes and merge them.

Input: DMA (Defense Mapping Agency) tapes. The tapes have Digital Terrain Elevation Data (DTED).

Output: raw files.

Programs: dtptopconv_sun4 (~irit/programs/topconv/dtptopconv_sun4), dtpunion_sun4 (~irit/programs/union/dtpunion_sun4).
dtptopconv_sun4: reads a tape and outputs raw files. One file for each cell (1 by 1 degree).
Help: look at ~irit/programs/topconv/dtptopconv.c
dtpunion_sun4: merge raw files. Output: 1 raw file.
Help: look at ~irit/programs/union/dtpunion.c

3.2 IIS SW

The IIS SW is the International Imaging Systems (IIS) System 600 software together with the Model 75 image processing unit (M75).

In order to use the IIS you have to work with the *trane* workstation. (Some IIS programs can be run on other computers).

The IIS SW has many image processing functions, is fast and the results can be displayed immediately on the M75.

3.3 IMG SW

Programs to manipulate img files and convert files to and from imgfiles.
The IMG programs run on any SUN workstation. Some programs (ice, imgcpt) need a color display.
Some IMG programs have similar IIS functions.

Programs: cpt2ps, iis2img, iis2ras, img2asc, img2ras, img2raw, imgcon, imgcpt, imgedit, imgext, imgscale, imgsetbounds, kdi, rasext, rasmask, raw2img.
stereo1(~irit/programs/stereo/stereo1/stereo1),
stereo2 (~irit/programs/stereo/stereo2/stereo2)

Manuals: Online: for each program type e.g.: man iis2img or: iis2img -H

3.4 ice

ice is a program that allows PostScript and rasterfiles to be composed into a merged image which may be both interactively previewed and saved to a file.

Manual: Online: Type: man ice

3.5 GMT SW, ikp

ikp is a program which allows a user to create and execute an arbitrarily connected network built from stand-alone programs (usually the GMT programs). Each stand-alone program is represented by an icon that has a popup panel.

The GMT programs are used for retrieval, processing, and display of "map-type" data (data which is referred to two coordinates).

Input to the GMT programs: ASCII files of (x, y) or (x, y, z) data.
x - longitude.
y - latitude.
z - intensity at (x, y).

The input is created in various ways: a digitizer and its associated programs, geobase and its associated programs, by-hand, etc.
These programs are not in the scope of this document. However, some of them are mentioned in chapter 4.10: Programs to convert files.

Output from the GMT and ikp: PostScript files.

GMT programs: grdcontour, psbasemap, pstext, psxy,
~irit/programs/new_psxy/new_psxy, ~irit/programs/new_psxy/psxygb,
/miles/image/irit/wsmith/c/img2grd, /miles/image/irit/wsmith/c/grd2img.

IMG programs: ikp.

Manuals: Online: Type: man ikp, gmtsystem, grdcontour, psbasemap, pstext, psxy
Type: new_psxy -H or: look at ~irit/programs/new_psxy/new_psxy.c
Type: psxygb -H or: look at ~irit/programs/new_psxy/psxygb.c
Type: img2grd -H
A hardcopy at the computer room: *The GMT System v.1.0,*
Technical Reference & Cookbook by Paul Wessel and Walter H.F. Smith

3.6 sun_sap

A program to print sun rasterfiles on the Iris printer.
See chapter 7.0 Printing.

4.0 DATA FORMATS

4.1 Topographic tapes

Resolution: 0 - 50 degrees: 1200 x 1200 pixels per degree.
50 - 70 degrees: 600 x 1200 pixels per degree.
70 - 75 degrees: 400 x 1200 pixels per degree.
etc.
The resolution is approximately 100 x 100 meter per pixel (or better).

Manual: A hard copy: *Defense Mapping Agency Product Specifications for Digital Terrain Elevation Data (DTED) - April 1986* (in Irit's office).

4.2 Spot tapes

Spot images are panchromatic or multispectral. Each Spot scene represents a ground area of 60 km x 60 km.

Resolution: pan - 10 x 10 meter or: XS (3 bands) - 20 x 20 meters.
1A - raw scan or: 1B - the pixels were corrected.

Manual: The tapes arrive with imaging and tape parameters and a histogram for each band (in Irit's office).

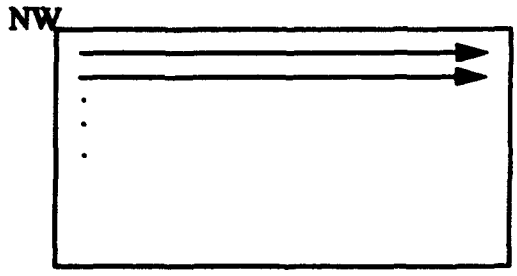
4.3 Raw files

Raw raster data.
Successive planes of binary raster data beginning with the front plane of the raster. Each raster plane is ordered by rows beginning with the top row.
The raster data may be of any one of several primitive data types such as byte, integer, real, etc.
So far, we have had only 1 plane files.

It is sometimes referred to as IIS convention, which is not an IIS format!

File names: *.raw, *.topo, *.conv

For 1 plane:



Manuals: Online: Type: man raw2img

4.4 IIS files

Iis files are image files used by the IIS SW. They are also referred to as System 600 format.
The IIS SW stores images as four-dimensional data sets, the dimensions being sample (x), line (y), band (z), and time (w). In addition, User Coordinate System (UCS) and statistical information can also be stored in the image.

The IIS SW creates and uses files other than image files. These files are identified by the file extension (or suffix) appended to the end of the file name. They will not mentioned further in this document.

File names: *.image

Manuals: hard copies: *Users Guide.*
Command Reference - CPU.
Command Reference - M75.
Command Reference - Utilities.
(in the computer room, by *trane*).

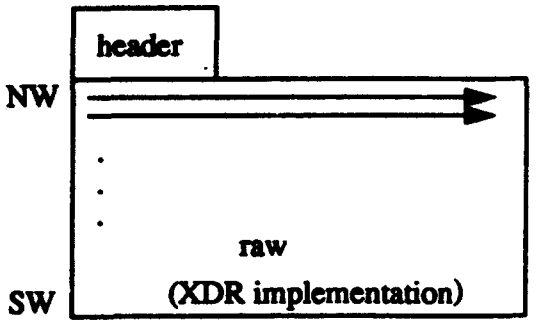
4.5 Img files

Used by the IMG SW. Designed at Lamont.

An IMG file is used for the storage of two- and three- dimensional binary raster data.
It is composed of a header followed by the raster data, which may be of any one of several primitive data types such as byte, integer, real, etc.
The raster data is stored a plane at a time from front to back in increasing z order. Each plane of data is stored a row at a time from top to bottom in increasing y order. Each row of data is stored from left to right in increasing x order. So far, we have had only 1 plane files.

File names: *.img

For 1 plane:



Manual: Online: Type: man imgfile

4.6 Color files

A color description file is used in conjunction with an IMG raster data file to create a 24-bit Sun rasterfile. This file completely specifies the RGB values for all data values within any img file which may be used in association with it. It is an ASCII text file.

File names: *.cpt

Example (for a file with data values in the range 5. - 1590.):

```
   5.  0   0   0   10. 25   0   1
  10.  25  0   1
    .
    .

1320.  0 255 255 1590.  255  0  0
```

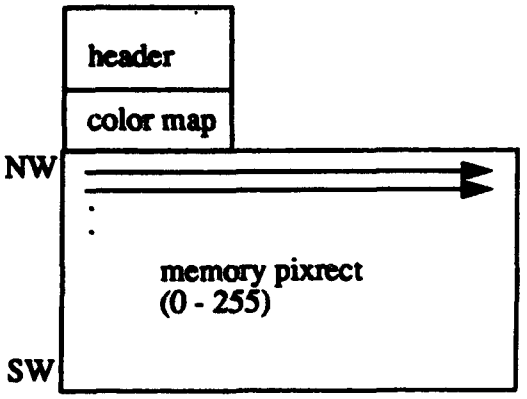
Manual: Online: Type: man colorfile

4.7 Rasterfiles

Sun's file format for raster images.
Used for printing and in some interactive display software.

A rasterfile is composed of three parts: a header, a (possibly empty) set of colormap values, and the pixel image, stored a line at a time. The image is laid out in the file as in a memory pixrect.

File names: *.ras, *.iris



Manuals: Online: Type: man rasterfile
Type: more /usr/include/rasterfile.h

4.8 PostScript (PS) files

A typical PostScript file has a beginning (features), a middle (commands to do plotting), and an end (tells the graphics device to put out the plot - "show page").

File names: *.ps, ps*.

Manuals: Online: Type: man gmtsystem.
Hard copies: *PostScript Language, Tutorial and Cookbook, Adobe Systems Inc.*
PostScript Language, Reference Manual, Adobe Systems Inc. (in the computer room).

4.9 ASCII files for GMT programs

These files are created via various programs and by an editor. The programs to create these files are not in the scope of this document. However, some programs are mentioned in chapter 4.10: Programs to convert files.

The ASCII files include (sometimes) a header followed by a list of longitudes and latitudes, and sometimes intensity or other values.

Check each GMT program for its input.

4.10 Programs to convert files.

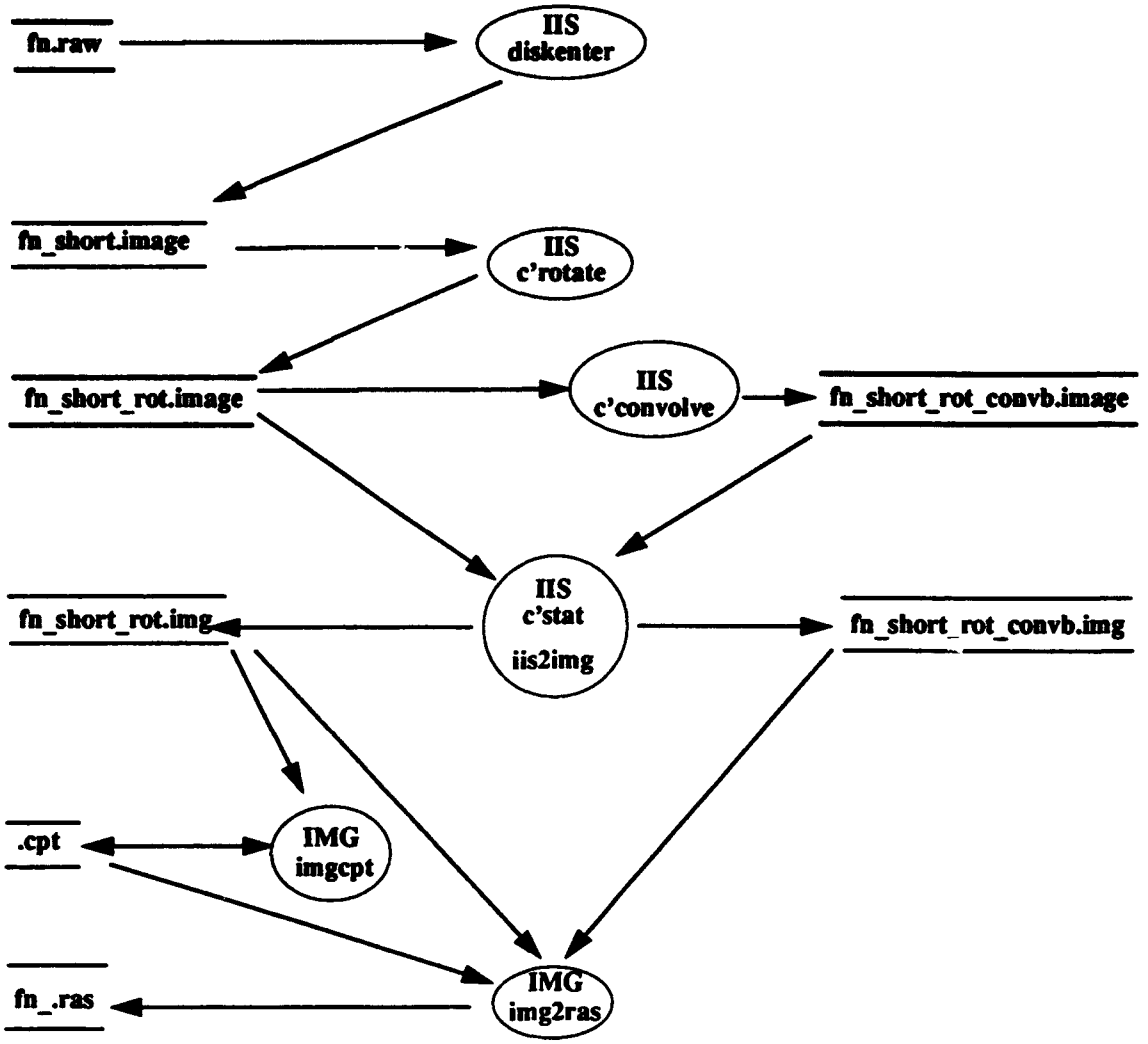
dtptopconv_sun4 (RAW SW): topographic tapes to raw files.
dtpunion_sun4 (RAW SW): merge raw files to one raw file.
spot (IIS SW): spot tape to iis file.
diskenter (IIS SW): raw to iis file.
disktransfer (IIS SW): iis to raw file.
raw2img (IMG SW): raw to img file.

img2raw (IMG SW): img to raw file.
iis2img (IMG SW): iis to img file.
imgext (IMG SW): img to img.
img2ras (IMG SW): img to rasterfile.
iis2ras (IMG SW): iis file to rasterfile.
rasext (IMG SW): rasterfile to rasterfile.
GMT SW: ASCII input files, grd files, etc. to ps files.
ice (IMG SW): ps, rasterfiles, and user symbols to ps or rasterfile.
sun_sap: rasterfile to a colored hard copy on the Iris.
lpr: ps, ASCII files to a hard copy on the laser printer (LW).

Additional programs (deal with GeoBase, grd files, ASCII files, etc.):
img2grd (GMT SW): img to grd file (a format used in gmt programs).
grd2img (GMT SW): grd to img file.
img2geo (~irit/programs/img2geo/img2geo): img file to GeoBase input files.
gb2gmt (~irit/programs/gb2gmt/gb2gmt): GeoBase ASCII output file to ASCII input file for GMT SW.
psxygb (GMT SW): GeoBase output ASCII file to ps file.
new_psxy (GMT SW): ASCII file (that "has" many ASCII files) to ps file.

5.0 Processing Topographic Data

The following diagram shows a sample of processing topographic data.



A raw file (fn.raw) is converted to an iis format (fn_short.image). This is done with the iis function diskenter. The data is of type short. Then, the data is corrected using the iis function c'rotate; x_scale_factor = cosine of the average latitude. The output is fn_short_-rot.image. This file is converted to an img file fn_short_rot.img.

Meanwhile, a gradient file is created using the iis function c'convolve with a kernel that determines the gradient (e.g. -1 0 0 0 0 0 0 0 1), the output is scaled to byte, from (-512, 512) to (0, 255). The output file is fn_short_rot_convb.image. This file is converted to an img file fn_short_rot_convb.img.

fn_short_rot.img is used to create a color file (.cpt), using the img function imgcpt.

The 3 files: fn_short_rot.img, fn_short_rot_convb.img, and .cpt are combined in the function img2ras to create a raster colored file with intensities.

6.0 Image processing with the IIS

The image processing system made by International Imaging Systems Inc. (IIS) is compound of the Model 75 Digital Image Processor (M75) and its System 600 Image Processing Software (IIS SW). The M75 is connected to a host computer (*trane*). It runs under the Unix operating system.

6.1 Reading

There is a set of manuals in the computer room.
Start by reading the *Users Guide*, especially the following chapters:
The Users Interface: chapter 3: *Command Syntax*, chapter 7.1, 7.2: *Image Files*, chapter 8: *System 600 on Unix.*
The Tutorial: chapter 5.1: *Image Input.* Chapter 6 in the Tutorial has a sample tutorial (in order to run it you have to rewrite the commands in Unix).
The Categorical List of Commands: pages 4,5 and the lists of commands.

While reading, refer only to the Batch Executive (BE) mode, the m75, system, tape, and cpu commands, and the Unix operating system.

When using the IIS look for each command in the *Command Reference.*

6.2 Basic Principles

There are m75, cpu, system and tape commands.

Most m75 commands can operate on either display images or disk images.
Disk images reside on the computer's disk. Display images are stored in Refresh Memory. They are visible on the display monitor (the M75 display screen).

There are 14 Refresh Memory (RM) planes, each of which can store 512 x 512 x 1 byte (= 256 Kilobytes in each RM).
Each RM is the equivalent of 1 black and white image (grey scaled) or 1 band of a color image. Color images are displayed using 3 RM together for red, green and blue.
12-bit images or images greater than 512 x 512 can be displayed using the m75'virtual'-roam command.
The contents of the RM are usually lost when you quit, so anything you want to keep must be saved as a disk image.

The M75 also has 8 graphics planes in memory, which you can use for overlays or to define a Region Of Interest (ROI). These are bit planes, 512 x 512 x 1 bit. When you are using an interactive command and step on the foot switch to see the button menu, you are looking at a graphics plane.

CPU commands operate only on disk files.

Use the M75 commands for visual, fast reference and then the cpu commands.

An IIS command may display a disk image (e.g, "v'roam"), load a disk image into the RM as a display image ("m'display"), manipulate a display image from RM ("m'scale") and write it into another RM or to disk ("save") or leave it in the pipeline (part of the M75), where it can be fed to RM using "feedback".

6.3 A Sample Session

1. Before starting the first time be sure to be a part of the *iis* user group.
2. Login to *trane*.
3. Change to the desired directory, type (boldface represents user input):

   % cd /trane/src/iis/deuser/iisimages

   % ls

   aust.image s575trans.dat sfo.image sfotemp.image topo.image
   kili.image sf81.image sfo.img text.image userlog.lis

4. Type, or put in your *.cshrc*:

   % source ~iis/usrenv

5. Invoke the Batch Executive(BE) by typing:

   % be

   1) s'dir

   Directory /trane/src/iis/deuser/iisimages/
   aust.image kili.image s575trans.dat sf81.image
   sfo.image sfo.img sfotemp.image text.image
   topo.image userlog.lis
   Total of 10 files.

   2) m'display sfo sf

   Loaded sfo into display image sf.

   [sf is displayed on the m75. sf is a display image.

   m' is an abbreviated form for m75.

   sfo is an abbreviated form for a disk image: sfo.image]

   3) m'scale sf

   [sf is scaled. The new image you see on the screen does not have a name and is not yet stored in RM].

   4) m'dir

   m750 - 11 Free RMs - 14 Total RMs
   sf - 512 x 512 x 3 , Byte

   5) feed -color sc

   [the scaled image is saved in a RM as a display image: sc]

   6) m'dir

m750 - 8 Free RMs - 14 Total RMs

sc - 512 x 512 x 3 , Byte

sf - 512 x 512 x 3 , Byte

7) m'histogram sf

8) v'roam sfo

[use the foot pedal to see what the buttons do]

9) sel sf(1)

[display only band 1]

10) m'del sc

11) quit

6.4 Trouble shooting

1. While in be mode, if you get the message:

Display management structure corrupted

Check:

) m'dir

Try:

) all m75

6.5 Some useful Commands

The *Users Guide* includes also a *Categorical List of Commands.*
Some useful commands are:
General management: m'dir, s'dir, s'quit
Image management: c'copy, m'display, m'save, m'del, m'dir, m'lock, m'unlock, m'feedback, c'diskenter, c'disktransfer,
Image input: tape'spot, tape'enter,
Image examination: m'roam, m'v'roam, m'flicker, m'split,
Image combination: c'merge,
Image statistics: c'histogram, m'histogram, c'statistics, m'statistics
Radiometric transforms: m'piecewise, m'histogram'equalize, c'scale, m'scale,
Spatial transforms: m'compass, c'compass, c'convolve, m'convolve, c'laplace, m'laplace, c'roberts, m'roberts, c'sobel, m'sobel, m'wallis,

7.0 PRINTING

7.1 PostScript file (fn.ps) on the laser printer (LW)

Type: **lpr fn.ps**

7.2 Rasterfile (fn.ras) on the Iris 3024

People: Kazuko, Russell, Irit.

1. Make sure to be part of the *Iris* users' group (one of the system managers will do it for you).
2. **rlogin trane**
3. **cd..** to the desired directory.
4. **all iris**
5. **sun_sap -c#:# -y#:# -m#:# -k#:# -h# -v# -r# [-n] [-G#] [-R#] [-s#] [-o#] fn.ras**

   -c#:# cyan contrast (50 - 250) : density (0 - 100)

   -y#:# yellow contrast (50 - 250) : density (0 - 100)

   -m#:# magenta contrast (50 - 250) : density (0 - 100)

   -k#:# black contrast (50 - 250) : density (0 - 100)

   -h# horizontal offset in pixels (0 - 7200)

   -v# vertical offset in pixels (0 - 7200)

   -r# pixel replication factor (1 - 8)

   -n disable automatic range expansion and Gamma correction (fastest)

   -G# Gamma correction. Default: 1. Has no effect if you use -n.
   Use: G = 0.7 to lighten the intermediate colors.

   -R# repeat image around the drum with # of pixels in between

   -s# the distance between images along the drum in pixels

   -o# overstrike each pixel # times (1 - 4).
   Use: -o2 for transparent paper.

   For a colored image use: -c250:50 -y250:50 -m250:50 -k250:50

   For a black and white image use: -c250:1 -y250:1 -m250:1 -k250:50

   Use either -n or -G0.7

   Gamma correction: x (colors: 0 - 1) are transformed to x**G.

   Papers: regular, glossy, or transparent.

   Resolution: 300 dots per inch.

   Size: up to 24 x 24 inches.
6. **deall iris**

8.0 Read Spot tape

Use the IIS SW command: *tape'spot*

Input: spot tape: pan (1 band) or xs (multispectral)

An example to a typical session:

1. **rlogin chaos** (or miles - Sun 4)
2. all 0
3. cd .. (change to the appropriate directory)
4. be

   1) **spot**

   tape'spot requires exactly 1 input operand

   Tape unit: **tape2**

   Output image: **garm_spot2** (name of disk image)

   [TYPE] Image type ((p) or xs)? **xs**

   [TAPE_NAME] Name of tape for operator to mount (SPOT)?

   [DIRECTORY_ONLY] Read and print directory only (NO)?

   [UPDATE_DB] Update the data base (NO)?

   [FORMAT] Format of tape ((BIL) or BSQ)?

   [WRITEHAAT] Write out a skeletal HAAT file for use with image-to-image correction? (N)

   (applies only to tapes created with System 600)

   [MAX_ERRS] Maximum number of parity errors to ignore (0)?

   Please mount tape SPOT on tape drive tape2

   Tape SPOT successfully mounted on tape2.

Last updated: January 30, 1992.

NAME

ikp - icon processing system

SYNOPSIS

ikp [-display display] [-b] [-B] [netfile] [-n] [-l libfile] [-v level]

DESCRIPTION

ikp is a graphical X11 interface which allows a user to create and execute an arbitrarily connected process network built from standalone programs. The program may also be executed outside of the X11 window system by running it in batch mode using a net configuration file which has been previously created in interactive mode.

Each standalone program, or process, is represented by a bitmap icon displayed in an X11 window. Each process may have any number of input and/or output connectors representing data streams which are to be read from or written to by the process. A connector is displayed as a small opening at the border of the process icon together with an arrow indicating the direction of data flow. Certain output connectors, known as diagnostic connectors, are displayed with a dashed arrow, and are generally used to output diagnostic error messages.

A group of processes may be linked together into a net by graphically connecting pairs of input and output connectors. Connectors may be bound directly to disk files as well as to other connectors. A net is deemed to be completely connected when all of its input and non-diagnostic output connectors have been either linked to other connectors in the same net or bound to disk files. The connection of diagnostic outputs is optional.

When a net has been completely connected, it may be executed, meaning that all processes in the net will be run as child processes of ikp with their inputs and outputs connected to each other via pipes and sockets. Output from unconnected diagnostic connectors is redirected to the controlling terminal of ikp, i.e., the window from which ikp was invoked. Individual processes within a net may be executed on remote machines as well as on the local workstation, enabling the user to more effectively distribute the computational load across a network.

In addition to supporting a small number of builtin process types representing the most commonly used input/output configurations, ikp maintains an internal database of both site-wide and user-specific program libraries, which are simply descriptions of external standalone programs. This database, which contains a full description of the command

syntax of each program in the library, allows ikp to present a more intuitive interface to the user and provide easy access to available software with which the user may be completely unfamiliar. Support for on-line help regarding individual programs is provided.

ikp could typically be used for such tasks as signal processing, seismic waveform manipulation, or any other procedure which requires that a number of standalone modules be interconnected to transform a group of input data sets into a group of output data sets. Unlike most command interpreters such as sh or csh, which only permit a linear connection of standalone programs via the stdin and stdout streams, processes within ikp can be connected in any arbitrarily complex topology, including feedback loops, via arbitrary data streams.

OPTIONS

-display display
Utilize the named X11 display. The default display is the first screen of the host on which ikp is invoked.

-b Operate in batch mode, reading the net configuration file specified on the command line and executing the process network described therein. The program will exit as soon as the process network runs to completion or any child process stops or exits abnormally.

-B This option is identical to the -b option described above, except that all child process stop and exit conditions are ignored.

netfile
Read in the process network described by the net configuration file netfile at initialization.

-n Do not use the default site program library file /usr/lib/ikp/sitelib. This file is normally read at initialization.

-l libfile
Read the named program library file at initialization. Any number of such library files may be specified as long as each one is preceded by the -l flag.

-v level
Specify a verbosity level for the printing of diagnostic messages to stderr. The default value of 0 suppresses all such output, while higher values produce a progressively greater volume of chatter. Specifying any positive value will, for example, display the full command lines of library processes whenever they are

constructed, a useful feature for the debugging of user-created program library files.

USAGE

When invoked in its usual interactive mode, ikp is driven by an X11 windowing interface. After initialization has completed, a large empty scrollable window is displayed within which the user may construct any number of nets. Access to all of the software's functionality is available via a pop-up menu which is displayed by depressing the right mouse button. This menu contains a number of submenus which are displayed by dragging the mouse to the rightmost edge of the main menu. The following paragraphs describe the purpose of all menu options.

Net -> Select. ikp is capable of manipulating any number of nets, and maintains a notion of a current net for certain purposes. A particular net can be made the current net either by selecting this option and then clicking the mouse over any process icon in the net, or by performing any editing operation on any process within the net.

Net -> Run. Execute the current net. While a process within the net is executing, its icon is displayed with a stippled light grey pattern. If an executing process becomes temporarily stopped for any reason, its icon will change to a stippled dark grey pattern. When a process completes, its icon will return to its original state. If a process is killed or stopped by a signal, or returns an exit status other than 0, its icon will be momentarily displayed in reverse video and the user will be prompted to take an appropriate course of action, e.g., ignore the event and proceed, attempt to restart the process if stopped, or abort all other processes in the net which are still running. Only one net may be run at a time, and a running net may not be edited in any manner whatsoever. (Operations on other nets may be performed while a net is running.)

Net -> Read. Read a net configuration file and build a new net from its contents. The upper left corner of the smallest bounding rectangle containing the entire new net will be placed at the location selected by the next mouse click. After the position is selected, the user is prompted for the name of the net configuration file to be read.

Net -> Write. Save the net containing the process icon selected by the next mouse click into a net configuration file. The user is prompted for the name of the file to be written. If the saved process network is functionally complete, it can be executed in batch mode at a later time by invoking ikp with the appropriate command-line option.

Net -> Copy. Copy the existing net selected by the next mouse click into a new net located at the position selected by a second mouse click. All user-specifiable values maintained within the edit panels of the existing net will be duplicated in the new net.

Net -> Stop. Temporarily suspend net execution by sending each running process within the net a SIGSTOP signal.

Net -> Restart. Resume net execution by sending each suspended process within the net a SIGCONT signal.

Net -> Abort. Terminate net execution by sending each running or suspended process within the net a SIGKILL signal.

Net -> Destroy. Destroy the entire net containing the process icon selected by the next mouse click.

Process -> Create. Create a new process and display its icon at the location selected by the next mouse click. When a process is created it is logically placed in a net by itself. As its connectors are linked to the connectors of other processes, its net is merged with the nets of said processes. A process is created with a fixed type of source, sink, filter, tee, custom or library.

Process -> Create -> Source. A source process has a single output connector which is mapped to its standard output and a single diagnostic connector which is mapped to its standard error.

Process -> Create -> Sink. A sink process has a single input connector which is mapped to its standard input and a single diagnostic connector which is mapped to its standard error.

Process -> Create -> Filter. A filter process has a single input connector which is mapped to its standard input, a single output connector which is mapped to its standard output, and a single diagnostic connector which is mapped to its standard error.

Process -> Create -> Tee. A tee process copies all data read from its single input connector onto each of its two output connectors.

Process -> Create -> Custom. A custom process has a user-specified number of input, output and diagnostic connectors. After the user clicks the mouse over the location at which the process icon is to appear, the user is prompted for the number of connectors. When the process icon is displayed, each connector is shown with the number of the file descriptor to which it is mapped. Custom processes are frequently

used to invoke ikp_cat(1), which concatenates an arbitrary number of input streams onto a single output stream.

Process -> Create -> [Library] -> [Module]. A library process is completely detailed by a module description in either the site program library file or some user-specified program library file. This description tells ikp what command is to be run, what the available command-line options are, which file descriptors are to be used for input and output connectors, etc. For each library known to ikp, a menu item is present within the Process -> Create menu which accesses a pullright menu containing menu items corresponding to each module description contained in the library. Selecting an item from this pullright menu will create a library process with the characteristics of the corresponding module. The library process type is intended as a method of providing easy access to software external to ikp without requiring detailed knowledge of that software's command syntax.

Process -> Edit. Clicking the mouse over a process icon after selecting this option causes the process to be opened for editing. A panel window is displayed which allows various attributes of the process to be interactively set by the user. The nature of these attributes is dependent upon the type of the process.

A command string, remote execution host, remote user, working directory and icon label may be specified for source, sink, filter and custom processes. The command string should contain the name of the program to be executed by the process together with any command-line arguments which that program may accept. Note that this string is similar but not identical to the string which you would type at a command interpreter such as sh or csh to execute the program, the difference being that ikp does not attribute any special meaning to such characters as <, >, *, $, etc. The only metacharacters recognized by ikp within a process command string are the single quote character ' , which may be used in pairs to enclose argument strings containing blank spaces, and the backslash character \ , which may be used to escape both the single quote and itself. (In other words, the behavior of these two special characters in this context is similar to their behavior under csh.) The icon label is displayed in the center of the process icon. If no icon label is specified, the last pathname component of the program name will be displayed in the icon.

A remote execution host and remote user may be specified for a tee process. Remote execution of a tee may be useful in minimizing network traffic when the tee is being used to connect remotely executing processes. The buffering mode to

be used by the tee may also be selected. The default memory buffering is almost always preferable, but special circumstances may require the use of file buffering (see ikp_tee(1)).

Library process edit panels allow the specification of a remote execution host, remote user, working directory, icon label and command path prefix. The name of the program to be executed is fixed, but its command-line arguments may be specified by making appropriate selections from whatever other panel items are present (these will vary depending upon the particular module description used to create the process.) A help button may be present to provide easy access to help information if such is available. If no icon label is specified, the program name will be displayed in the icon. The command path prefix can be used to specify the execution of some alternate version of the program, or to specify an absolute pathname for an executable which is not located in a directory contained within the user's search path.

If the working directory of a process is not specified, the process will be run in the directory in which ikp was invoked. (This applies to both local and remote processes.)

Remote execution involves a number of issues such as machine architecture and user environment. These questions are discussed in detail in a later section of this document.

Process -> Copy. Copy the existing process selected by the next mouse click into a new process located at the position selected by a second mouse click. All user-specifiable values maintained within the edit panel of the existing process will be duplicated in the new process.

Process -> Move. The display location of a process icon may be altered by clicking the mouse over its icon after selecting this option. The icon may then be dragged by the mouse. Click the right mouse button to fix the location and return to the main menu.

Process -> Load. Read a process library file and append appropriate pullright menus to the existing Process -> Create menu to enable the creation of library processes corresponding to module descriptions contained within the file. The user is prompted for the name of the file to be read. (The format of this file is described below.)

Process -> Stop. Temporarily suspend execution of the running process selected by the next mouse click by sending it a SIGSTOP signal.

Process -> Restart. Resume execution of the suspended process selected by the next mouse click by sending it a SIGCONT signal.

Process -> Abort. Terminate execution of the running process selected by the next mouse click by sending it a SIGKILL signal.

Process -> Destroy. Destroy the process selected by the next mouse click.

Connector -> Link. Link the two connectors selected by the next two mouse clicks. Note that an input connector may only be linked to an output connector and vice-versa. There is no explicit prohibition against linking two connectors of the same process.

Connector -> File. Bind the connector selected by the next mouse click directly to a disk file. A popup window will appear in which the user may enter the name of the file. If the selected connector is an output connector, the user may also specify whether the file, if preexisting, should be rewritten from the beginning or extended from its current end-of-file by toggling a truncate/append switch. A file which is bound to a connector of a remote process will be opened and accessed by the process on the remote machine - if you want a remote process to access a local file, the connector of the remote process should be linked to a local process which uses a command such as cat(1) to read the file and transmit it to the remote process.

Connector -> Ground. Ground the connector selected by the next mouse click directly to /dev/null, the system's virtual trashcan. This option is useful for discarding unwanted output from a process.

Connector -> Unlink. Break the link, file binding or ground to the connector selected by the next mouse click. Removing a connector link may cause a net to be divided into two disjoint nets.

Connector -> Edit. Change the file decriptor and/or filename and truncate/append mode mapped to the connector selected by the next mouse click. A file descriptor edit may be performed only on the connector of a custom process. A filename or truncate/append mode edit may be performed only on a connector which has been either bound to a disk file or grounded.

Connector -> Move. The display location of a connector may be altered by clicking the mouse over it after selecting this option. The connector may then be dragged by the mouse

to any point on the perimeter of its process icon. Click the right mouse button to fix the location and return to the main menu.

Directory. Change to a new working directory.

Redisplay. Redisplay the contents of the network window.

Exit. Exit the program.

ACCELERATOR FUNCTIONS

While the entire functionality of ikp is accessible through its menu system, some accelerators exist to more quickly invoke a small number of the most commonly used functions.

Clicking the left mouse button over a process is equivalent to invoking the Process -> Edit menu option on the selected process.

Clicking the left mouse button over a connector is equivalent to invoking the Connector -> Edit menu option on the selected connector.

Clicking the middle mouse button over a connector is equivalent to invoking the Connector -> Link menu option on the selected connector.

Clicking the middle mouse button over a connector while simultaneously depressing the keyboard Shift key is equivalent to invoking the Connector -> File menu option on the selected connector.

Clicking the middle mouse button over a connector while simultaneously depressing the keyboard Control key is equivalent to invoking the Connector -> Unlink menu option on the selected connector.

REMOTE EXECUTION

Any individual process within a net can be made to execute on a remote machine by specifying that machine's name in the remote host field of the process edit panel. It is also possible to cause the remote process to be run under an arbitrary user name by similarly specifying the remote user name. (An empty user field will cause the process to run under the local user name.) There are three issues relating to remote execution that must be kept in mind: (i) remote host availability, (ii) user authentication and permissions, and (iii) remote execution environment.

A process may be remotely executed only on a host which has access to the ikpd(8) remote execution service daemon. This program is normally started up by inetd(8). Its sole

function is to service remote process execution requests from users who are running ikp on another machine on the network.

The ikpd service daemon running on the remote machine on which process execution is being requested performs a user authentication check before granting execution privileges. This authentication check is similar to that performed by rlogin(1) and rsh(1), i.e., (i) a lookup of your user name in the remote password database is performed on the remote machine to determine if you have login privileges, and (ii) the .rhosts file in your home directory on the remote machine is scanned for the hostname of the machine from which you are requesting remote execution (in other words, the machine on which you are running ikp). If either of these steps fail, remote execution privilege is denied. If you have requested remote execution under a remote user name which is different from your local user name, (i) a lookup of the remote user name in the remote password database is performed on the remote machine to determine login privileges, and (ii) the the remote user's .rhosts file is checked to ensure that access from your local user name and local host has been specifically granted by scanning for a line whose first token is the hostname of the machine from which you are requesting remote execution, and whose second token is your local user name.

If the remote authentication check succeeds, ikp (and the cooperating remote ikpd) will attempt to construct a remote execution environment for you before actually running your remote process. This execution environment is a list of environment variable names, together with their values, which will be inherited by the remote program which you want to run. You can generate a complete list of all your local environment variables by using the printenv(1) command in any shell window. One environment variable, the PATH variable, is absolutely critical to the successful use of ikp's remote execution facilities, as it is used to locate the actual executable program file which you are attempting to run (unless you have specified an absolute pathname as the command name of a source, sink, filter or custom process, or as the command path prefix for a library process). ikp cannot simply use your local PATH variable on the remote machine because (i) executable programs may be stored in different locations on different machines, and (ii) the location of the program may be dependent upon the architecture of the remote machine, which may be different from the architecture of the local machine.

The remote execution environment is constructed as follows: (i) your entire local environment is copied to the remote machine, (ii) the remote machine architecture is determined,

e.g., sun3, sun4, etc., and is used to set (or replace) the value of the ARCH environment variable, (iii) any instance of the string IKPRARCH which occurs in any environment variable value is replaced by the new value of the ARCH environment variable, (iv) the name of your home directory (or that of the remote user name you have specified, if any) on the remote machine is determined, and is used to set (or replace) the value of the HOME environment variable, (v) any instance of the string IKPRHOME which occurs in any environment variable value is replaced by the new value of the HOME environment variable, and (vi) if the environment variable IKPRPATH exists, its value will be used to set (or replace) the value of the PATH variable.

It is therefore possible to dynamically adjust your PATH variable to appropriate values for different remote machines by including a line in your .cshrc file such as the following:

```
setenv                                                  IKPRPATH
.:IKPRHOME/IKPRARCH:/usr/local/IKPRARCH:/usr/ucb:/bin:/usr/bin
```

Then, if you request remote process execution on a machine of architecture type sun4 where your home directory is /home/elvis, your ARCH, HOME and PATH environment variables will be set as follows before the remote process is executed:

```
ARCH=sun4
HOME=/home/elvis
PATH=.:/home/elvis/sun4:/usr/local/sun4:/usr/ucb:/bin:/usr/bin
```

PROGRAM LIBRARY FILE FORMAT

ikp can be informed of numerous characteristics of an external program by reading in a program library file which contains a description of the program. A library is defined as a collection of modules, and a module is defined as a description of a program and its command syntax.

Many of the lines in a library file will be of the form

keyword value

where keyword is some attribute and value is the value of that attribute. Double quotes may be used to indicate a null value or to include whitespace characters within the value, e.g., "" (the null string) and "This is a string containing whitespace." In general, declaring a value to be "" is equivalent to omitting the declaration line entirely. All blank lines and lines whose first non-whitespace character is # are ignored. Tabs and blank spaces may be used at will anywhere within a line to improve readability by indenting, aligning, etc.

A library is fully described by the lines

```
library name {
    module
```

```
        module
        ...
}
```

i.e., by an identifying name together with a sequence of module decriptions. The supplied name will be the string inserted into the Process -> Create menu.

A module is fully described by the lines

```
module name {
        program    executable-name
        helpfile        helpfile-name
        pathprefix      executable-path-prefix
        directory  working-directory
        host       execution-host
        input           input-fd
        ...
        output          output-fd
        ...
        diagnostic      diagnostic-fd
        ...
        item
        item
        ...
}
```

i.e., by an identifying name together with several key/value pairs and a sequence of item decriptions. The supplied name will be the string inserted into the Process -> Create -> [Library] menu.

The program entry specifies the name of the executable command that will be invoked when the process is run. This is the only key/value pair that must be explicitly set within a module description to a non-null value. This value cannot be changed by the user from the process edit panel.

The helpfile entry specifies the name of an ASCII text file containing help information for the module. If this is a non-null string, the process edit panel will contain a help button which will cause the contents of the named file to be displayed when it is selected.

The pathprefix entry specifies a path to be prefixed to the program name. This can be useful if the user's search path does not contain the command specified by the program entry, or if the user wishes to run some alternate version of the command.

The directory entry specifies the working directory in which the command is to be executed.

The host on which the command is to be executed may be specified with the host entry.

The input, output and diagnostic entries declare the file descriptors which are to be mapped onto the process input, output and diagnostic connectors. There may be any number of these entries. When the process icon is displayed, each connector is shown with the number of the file descriptor to which it is mapped. Ideally, the help file should document the nature of each such descriptor so that the user can fully understand the purpose of all displayed connectors.

Item descriptions are used to control the specification of command options. Each item description corresponds to a panel item which is built into the process edit panel. When the user interactively changes the value of the panel item, the command-line argument list with which the command is invoked is rebuilt to reflect the changes made by the user. Setting a panel item to a certain value may cause other items in the edit panel to be enabled (i.e., made visible and sensitive to user input) or disabled. An item description may contain other item descriptions, generally those which it is capable of enabling and/or disabling.

There are four types of item descriptions, choice, cycle, toggle, text and slider, corresponding exactly to the Lamont X Toolkit (LXT) panel item types. See the LXT documentation for a description of these. Typically, the first three types are used to set option flags and enable/disable other items, the text type is used to allow user editing of text strings such as filenames, and the slider type is used to allow specification of an integer within a fixed domain.

ikp builds the command-line argument list to the declared executable program in the following sequence: (i) zero the character string used to hold the command, (ii) append any pre-declared or user-edited executable path prefix (if any), (iii) append the name of the executable program, and (iv) for each enabled item in the order in which they have been declared in the module description, append arguments appropriate to the particular item. Step (iv) is performed according to rules which are particular to each item type and which are explained in full detail below. Note that if an item is disabled, all of its subitems are considered to be disabled.

Each item description is of the form

item *type* *ID-number* {
    declaration
    ...
}

where *type* is one of choice, cycle, toggle, text or slider, and *ID-number* is an integer identification tag associated with the item. The first item in a module must have a tag of 0, and subsequently declared items (including nested item

descriptions) must have consecutively increasing tag values.

Items of any type may contain any of the following declarations:

column	m
row	n
xoffset	x
yoffset	y
string	item-label
option	item-option-string
prevoptcontig	

The column, row, xoffset and yoffset entries are used to position the item in the process edit panel. The column and row entries are translated into a pixel location in the panel using a formula based upon the height and width of the font being used. This location is then adjusted more finely by applying the pixel offsets declared by the xoffset and yoffset entries. The default value of 0 for all of these entries will cause the item to be located just beneath the module name, path prefix, working directory and execution host items which always appear at the top of any library process edit panel.

The string entry is used to set the LXPI_STRING attribute of the panel item.

The option entry declares an item-option-string which will be inserted into the command-line argument list if this item is enabled at the time the list is built.

If the prevoptcontig declaration is present, the item-option-string will be appended to the existing command-line argument list with no intervening white space between the item-option-string and any preceding argument (or between the item-option-string and the command itself if there are no preceding arguments).

Text items may contain any of the following declarations in addition to the universal item declarations described above (note that they may not contain any nested item descriptions):

value	value-string
store	max-stored-value-length
display	max-displayed-value-length
nonnull	
integer	
real	
encapsulate	
optvalcontig	

The value, store and display entries are used to set the LXPTEXT_VALUE, LXPTEXT_MAXSTORE and LXPTEXT_MAXDISPLAY attributes of the item. The value-string may be interactively changed by the user to any string whose length does not exceed max-stored-value-length.

The nonnull, integer and real entries will cause an error to be reported to the user at the time the net containing the process is executed if the value-string of the item does not satisfy the stated condition, i.e., if the value-string is null, is not an integer, or is not a real number.

The encapsulate entry will cause the item's value-string to be enclosed within a pair of special encapsulation characters when it is appended to the command-line argument list. This forces the command which is ultimately run when the enclosing net is executed to treat the value-string as a single argument regardless of any blank spaces which it may contain. This is particularly useful when the value-string represents a file name, which on many systems may legally contain a blank space. The encapsulate declaration should be made for any text item whose value-string is used to construct an argument that might meaningfully contain blank space. The internal encapsulation character currently used by ikp is the tab character, which means that tab characters should not be present in any item-option-string, value-string or selection-option-string (see below) declared within a program library file. Any such character will be replaced by ikp with a single blank space.

If the optvalcontig declaration is present, the item's item-option-string and value-string will be appended to the existing command-line argument list with no intervening white space between the item-option-string and the value-string.

ikp will append text item option flags and/or other arguments to the command-line argument list in the following way: (i) append a blank space unless a prevoptcontig declaration is in effect for the item, (ii) append the declared item-option-string (if any), (iii) append a blank character unless an optvalcontig declaration is in effect for the item or the value-string is empty, and (iv) append the item's value-string (if any). If both the item-option-string and the value-string are empty, nothing is appended to the argument list.

Slider items must contain the following declaration in addition to the universal item declarations described above:

range min-value max-value

The integers min-value and max-value are used to set the LXPSLIDER_MINVAL and LXPSLIDER_MAXVAL attributes of the item.

Slider items may also contain any of the following declarations (note that they may not contain any nested item descriptions):

value value
barlength barlength
optvalcontig

The value and barlength entries are used to set the LXPSLIDER_VALUE and LXPSLIDER_BARLENGTH attributes of the item. The value may be interactively changed by the user to any integer between min-value and max-value.

If the optvalcontig declaration is present, the item's item-option-string and value will be appended to the existing command-line argument list with no intervening white space between the item-option-string and the value.

ikp will append slider item option flags and/or other arguments to the command-line argument list in the following way: (i) append a blank space unless a prevoptcontig declaration is in effect for the item, (ii) append the declared item-option-string (if any), (iii) append a blank character unless an optvalcontig declaration is in effect for the item, and (iv) append the item's value.

Choice, cycle and toggle items are all variants on the enumerated item type. (See the Lamont X Toolkit documentation for further explanation.) Enumerated items may contain any of the following declarations in addition to the universal item declarations described above:

format format-type
value value
selection
...
item
...

The format entry specifies a format-type which must be either H or V. This declares the visual layout of the item's selections to be either horizontal (the default) or vertical.

The value entry, which may be interactively altered by the user, sets the LXPENUM_VALUE attribute of the item.

The item must contain at least one selection declaration and may contain any number of nested item declarations (which may in turn contain other nested declarations). Selection

declarations correspond to the cardinal-valued selections of the item. Each such declaration is of the form

selection ID-number {
    declaration
    ...
}

where ID-number is an integer identification tag associated with the selection. The first selection in an item must have a tag of 0, and subsequently declared selections must have consecutively increasing tag values. The ID-number is considered to be the value of the selection. If the value of a choice or cycle item is n, then that selection with value n is considered to be active, and all other selections are considered to be inactive. If the value of a toggle item is n, then a selection within that item is active if and only if its value s satisfies the condition (n & (0x1 << s). Note that a choice or cycle item must have exactly one active selection, while a toggle item may have any number of active selections.

A selection may contain any of the following declarations:

string	selection-label
option	selection-option-string
prevoptcontig	
enable	ID-number
...	
disable	ID-number
...	

The string entry sets the LXPENUM_SELSTRING attribute of the selection.

The option and prevoptcontig entries are used during the construction of the command-line argument list, which is done by ikp in the following way: (i) append a blank space, unless a prevoptcontig declaration is in effect for the item or the item-option-string is empty, (ii) append the declared item-option-string (if any), and (iii) for each active selection within the item, (iii-a) append a blank character, unless a prevoptcontig declaration is in effect for the selection or the selection-option-string is empty, and (iii-b) append the selection's selection-option-string (if any). If the item-option-string and all active selection-option-strings are empty, nothing is appended to the argument list.

The enable and disable entries specify the integer ID tags of other items (usually nested sub-items) which are to be enabled and/or disabled when the selection is activated. Additionally, if the item is a toggle item and the selection becomes deactivated, all items in its enable list will be disabled and all items in its disable list will be enabled.

The enabling and disabling of items are recursive procedures. If an item with enabled subitems becomes disabled, all of its enabled subitems become disabled. If the item is enabled again at a later time, all of its previously enabled subitems will once again be enabled.

The following example and subsequent description are provided to further illustrate the techniques of constructing a program library specification file.

```
#
# Seismic Processing software
#
library "Seismic Processing" {

     module "Instrument Response" {

          program        ahinstr
          helpfile       /usr/lib/ikp/ahinstr.help
          input          0
          output         1

          item choice 0 {
               column          0
               row        0
               format          H
               string          Mode
               value           0
               selection 0 {
                    string          Insert
                    option          -i
               }
               selection 1 {
                    string          Remove
                    option          -r
               }
          }

          item choice 1 {
               column          0
               row        1.5
               format          H
               string          Filter
               value           0
               selection 0 {
                    string          None
                    option          ""
                    disable         2
                    disable         3
               }
               selection 1 {
                    string          "High pass"
                    option          -h
```

```
                enable          2
                enable          3
            }
            selection 2 {
                string          "Low pass"
                option          -l
                enable          2
                enable          3
            }
            item text 2 {
                column          0
                row         3
                string          Roll:
                option          ""
                store           20
                display         10
                value           ""
                real
            }
            item text 3 {
                column          20
                row         3
                string          Zero:
                option          ""
                store           20
                display         10
                value           ""
                real
            }
        }

        item cycle 4 {
            column          0
            row         4.5
            format          H
            string          Taper
            value           0
            selection 0 {
                string          Disable
                option          ""
                disable         5
            }
            selection 1 {
                string          Enable
                option          -f
                enable          5
            }
            item text 5 {
                column          0
                row         6
                string          Filename:
                option          ""
```

```
                    store           120
                    display         30
                    value           ""
                    nonnull
                    encapsulate
               }
          }
          item slider 6 {
               column          0
               row        7.5
               string          Order:
               range           0 10
               value           0
          }
     }
}
```

The above example describes a library of seismic processing software containing a single program whose purpose is to modify instrument response. The actual executable is ahinstr. It reads input from file descriptor 0 and writes output to file descriptor 1. Help information related to this program is stored in the ASCII text file /usr/lib/ikplib/ahinstr.help.

Standard interactive shell usage syntax for this program is ahinstr [-i | -r] [-l roll zero | -h roll zero] [-f file] order
Note that the -i (insert response) and -r (remove response) options are mutually exclusive, as are the -l and -h options (low pass vs. high pass filter). Exactly one of the -i and -r options must be specified. Usage of the -l or -h option is not required, but if either option is used two additional numeric arguments must also be specified. The optional -f argument requires the specification of a single non-null text string. The required order argument must be an integer between 0 and 10.

Item 0, a choice item, allows the user to select between the mutually exclusive operations of response insertion and removal. The declared item value of 0 initializes selection 0 to be the active selection. If the user does not alter the value of the item by clicking the mouse over selection 1, ikp will insert selection 0's -i selection-option-string into the command-line argument list. If the user does activate selection 1, then selection 1's -r selection-option-string will be used. Since this is a choice item, exactly one of the two selections will always be active, so the argument list will always contain either -i or -r but not both.

Item 1, also a choice item, controls the program's filter options. If selection 0 is active (the declared initial state of the item), no option is inserted into the argument list (note the null selection-option-string which has been specified), and text items 2 and 3, which are used to set the roll and zero arguments, are disabled since these arguments will not be used. If selection 1 is activated by the user, the -h selection-option-string is inserted into the argument list, and text items 2 and 3, which are now required, are enabled. If selection 2 is activated by the user, the -l selection-option-string is inserted into the argument list, and text items 2 and 3, which are required, are enabled. If enabled, text items 2 and 3 are each required to have as their value-string a string which can be transformed into a real number. If, for example, the user edits one of these items to set its value-string to "hello", ikp will not allow the net containing the process to be executed.

Item 4, a cycle item, allows the user to optionally perform a tapering operation using parameters which have been stored in a file. If selection 0 is active (the declared initial state of the item), no option is inserted into the argument list (note the null selection-option-string which has been specified), and text item 5, which is used to specify the filename argument, is disabled since it will not be used. If selection 1 is activated by the user, the -f selection-option-string is inserted into the argument list, and text item 5, which is now required, is enabled. If enabled, text item 5 is forced to be a non-null string, and is encapsulated so that a user can specify a file name containing blank spaces if desired.

Item 6, a slider item, allows the user to choose any integer between 0 and 10 for the required order argument.

INSTALLATION

An attempt is made to read a site program library file at startup time. This file should be installed as /usr/lib/ikp/sitelib.

Tee processes are implemented via the ikp_tee program, which must be installed in a directory contained within the user's executable search path in order to be accessible by ikp.

ikp queries the network services database to determine the port number for remote execution service. This means that (on a Sun) an entry such as

ikp 8000/tcp

must be created in the /etc/services file. The port number (8000 in the above example) must be unique across all entries in the file, and must be greater than 1024. If ikp

is to be run on a network which utilizes the Yellow Pages network database lookup service, the edit should be made to the /etc/services file on the master YP server, and the services map should be rebuilt. If not, the edit should be made to the /etc/services file on every machine on which either ikp or ikpd will be run. (The same port number must be used in every /etc/services file).

The ikpd remote execution service daemon must be invokable by inetd(8) or already running with root privilege on each machine on which a user wants to run a remote process. It is usually started up indirectly via inetd by the presence of an appropriate entry in the inetd.conf(5) file such as

```
ikp     stream   tcp      nowait   root     /usr/etc/in.ikpd
in.ikpd
```

(In this example, ikpd has been renamed in.ikpd and installed in /usr/etc.) It may also be invoked directly at system boot time as a background process from /etc/rc. The following lines, placed near the end of that script, are one example of how this may be done.

```
#
# icon processor remote execution service daemon
#
if [ -f /usr/etc/ikpd ]; then
     /usr/etc/ikpd 2>/dev/console
     echo 'Starting ikpd ...' >/dev/console
fi
#
```

Note that the file system in which ikpd is installed must be mounted at the time these commands are executed (/ and /usr are generally safe choices - in the above example, ikpd has been installed in /usr/etc.)

XDEFAULTS

ikp attempts to determine user preferences for the following X11 resources (program defaults are shown in parentheses):

Background
Set the background color (White).

Foreground
Set the foreground color (Black).

Border
Set the border color (Black).

BorderWidth
Set the border width in pixels (2).

Font Set the font (8x13).

Geometry

Set the root window size and position (=600x600+100+100).

FILES

/usr/lib/ikp/sitelib Default site program library file.

BUGS

Because the inter-process communication links used by ikp are based upon pipe(2) and socket(2) facilities, user programs which are run from ikp must be prepared to handle partially successful read and write requests, e.g., an attempt to read n bytes may yield less than n bytes due to pipe buffer sizes, network transmission delays, etc. This is especially critical with regard to remote process execution. It will generally be safer to use buffered (rather than unbuffered) I/O when writing such programs.

There should be a more flexible mechanism for individually tailoring remote execution environments.

Diagnostic output from remote processes which is being redirected to the controlling terminal of ikp is handled by a buffering algorithm which may cause some output to be lost if the buffer is overrun. This problem should be exhibited only by programs which generate a continuous stream of diagnostic output.

SEE ALSO

ikp_tee(1), ikp_cat(1), ikpd(8)

AUTHOR

Roger Davis, October 1989.

NAME

ice - image composition environment

SYNOPSIS

ice [-display display] [-w width] [-h height] [-dpi resolution] [ice file] [-H] [-ps < ps_file]

DESCRIPTION

ice allows collections of PostScript and raster images to be composed into a merged image which may be both interactively previewed and saved to a file. The program operates under the Sun X11/NeWS window system on both color and monochrome displays. (Monochrome displays will of course result in some limitations in functionality, but may still be quite useful in many instances.)

OPTIONS

-display display
Use the named X11 display device.

-w width
Specify the width in inches of the composite image.

-h height
Specify the height in inches of the composite image.

-dpi resolution
Specify the resolution in dots per inch of the composite image.

ice file
Initialize the composite image and all program defaults by reading the named ICE file at startup time. (This file must have been created via one of the program's Dump options at some previous time.)

-H Display help information.

-ps Initialize the composite image by reading an external PostScript document from stdin at startup time. The contents of stdin will be stored in a temporary file named 'StdInput' in either the current directory, or, if the environment variable ICETMPDIR has been set, in the directory $ICETMPDIR.

USAGE

ice always displays its page window, which is used to show the user's composite image. Other secondary windows are displayed from time to time, generally in response to various user requests to modify some attribute of the composite image or one of its components. A popup menu which provides access to all of the program's functionality may be invoked

by depressing the right mouse button while the cursor is positioned over the page window.

There are two basic classes of external graphical objects which may be imported into ice - single-page PostScript documents and Sun rasterfiles, both of which must reside in pre-existing files. ice can also manipulate several classes of internally created PostScript-based graphical objects, including text, vectors, curves, markers, rectangles, polygons and axes. All of these external and internal objects are maintained on a single list which is ordered by sequence number.

Each object has its own sequence number, which is initialized to 0 but may be changed by the user to any other integer (including negative integers) at any time. When the composite image is rendered on the display or dumped to a file, objects with higher sequence numbers will be drawn after, and therefore overlay, objects with lower sequence numbers. There is no guarantee of the order in which objects with identical sequence numbers will be drawn.

Any collection of these individual objects (known as atomic objects) may be combined into a single composite object, thus enabling various operations to be performed on the group as a whole. Composite objects may be further grouped into larger composite objects.

In addition to the aforementioned classes of graphical objects, there is a special class of non-visible objects known as path objects. This class is essentially an implementation of the PostScript path concept, and is available for the control of various aspects of the display of certain visible objects.

The following paragraphs describe the purpose of all menu options. Note that in many cases, operations which modify an existing object may be initiated either by referring to the object's name as explicitly listed in the menu system or by selecting the object by pointing at it with the mouse and clicking any mouse button. Object selection is made by choosing the object whose origin location is closest to the location pointed to by the mouse. (Composite objects have no origin location of their own and are therefore selected by selecting one of their atomic children.) Also note that most user-specifiable dimensions are in points. (The point is a commonly used typographic unit of measurement equal to 1/72 of an inch.)

Insert. This option invokes a pullright menu which allows the user to insert various types of graphical objects into the composite image. In general, the insertion process

involves specifying various attributes of the object within a temporary popup window and then locating the object within the image. The latter operation is performed in one of two ways depending upon the state of the global location mode (specifiable via the Attributes -> Page menu option). If cursor location is enabled, the user must set the location by pointing and clicking the mouse. This is the default mode, and the details of this style of object location are described below for each of the various object types. If text location is enabled, the instructions for object location in the following Insert -> [object-type] options should be ignored - the user will be prompted to enter as many coordinate values as necessary via the keyboard into a temporary popup window. Certain types of objects require the specification of a filename from which the object is retrieved. The user may set the ICEDOCPATH environment variable to a colon-separated list of directories which will be searched for such files. If this environment variable is not set, only the current directory will be searched. All attributes of new objects are initialized to reasonable default values, but may be initialized otherwise via the Attributes -> Insert menu option as described below.

Insert -> PS Document. This option will load the contents of an external pre-existing PostScript file into the image. A temporary popup window is displayed which allows the specification of a filename, horizontal and vertical scaling factors, rotation (in degrees counterclockwise from the horizontal), clipping path, a dump transparency key (see the description of the Dump -> All -> Raster option below) and a sequence number. After the popup window is exited, the user must fix the origin location of the document's coordinate system within the composite image by moving the mouse cursor to the appropriate position and then clicking any mouse button. It is frequently desirable to set the origin location of the document to the lower left corner of the composite image - this can be easily done by depressing either the <SHIFT> or <CONTROL> key while clicking any mouse button irrespective of the position of the mouse cursor. An external PostScript file is essentially opaque as far as ice is concerned, i.e., it is rendered as an atomic entity - ice makes no attempt to extract any kind of graphical structure from the contents of the file. It is therefore impossible to interactively manipulate any portion of such external files.

Insert -> Raster. This option will load the contents of a 1-, 8-, 24- or 32-bit Sun rasterfile into the image. A temporary popup window is displayed which allows the specification of a filename, pixel replication factor, orientation, display mode, and foreground and background colors. Pixel replication allows the raster to be scaled by any integral multiple. (The default value of 1 will cause each pixel of

the original raster to be rendered as a single pixel in the composite image. A value of 2 would double the width and height of the original raster, e.g., each pixel of the original raster would be rendered as four pixels in the composite image.) Orientation is specified by setting the Raster Origin value to either Upper Left (the default), Lower Left (90 counterclockwise rotation), Lower Right (180 rotation) or Upper Right (90 clockwise rotation). Setting Display Mode to Full will cause the raster to be fully loaded into memory and shown on the display, while setting it to Outline will result in the display of a transparent rectangle which simply indicates the location and dimension of the raster. (Note that color rasters can be shown on a monochrome display only in outline mode, and that 24- and 32-bit rasters, when shown on a color display, will be rendered in 8-bit color using an internal histogram-based algorithm.) If the raster is a 1-bit (monochrome) raster, the foreground and background colors respectively corresponding to the 1 bits and 0 bits in the raster can be set to any combination of RGB primary intensities, which may range from 0 to 255. (The default settings are black foreground and white background.) After the popup window is exited, the user must fix the location of the raster within the composite image by moving the mouse cursor to the appropriate position and then clicking any mouse button. Note that ice is extremely memory-intensive, particularly when raster objects are heavily used. Performance can often be improved by loading raster objects in outline mode and switching to full draw mode to completely render them only when absolutely necessary.

Insert -> Text. This option will create an internal text object which may be used to annotate the composite image. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference via ice's menu system. The Source of the object may be set to either Interactive (which permits the user to enter a single line of text into the Text value) or File (which allows for the specification of a Filename from which a larger amount of multi-line text can be read). The family and name of the font to be used in rendering the text may be selected from a domain of available fonts which is constructed by examining the user's FONTPATH and OPENWINHOME environment variables as described in Sun's OpenWindows documentation. The text size and leading may be specified in points. (The specified leading is additional rather than absolute, e.g., to create 10 on 12 type, Size should be set to 10 and Lead should be set to 2. Reverse leading is possible via the specification of negative lead values.) Justification mode may be set to Flush Left, Flush Right, Center, Justify or Path. If path mode is used, the name of an

existing path object along which the text will be drawn must be supplied, and Path Offset and Letterspace values in points may also be given. Foreground and background colors can be specified. (The background is Transparent by default, but may be set to Opaque, which will result in the text being drawn upon a minimally bounding rectangular background of an arbitrary color. An opaque background may not be used with text which is rendered in path mode.) Horizontal and vertical scaling factors may be specified to create condensed or extended text. The Rotation value may be used to specify an arbitrary counterclockwise rotation in degrees from the horizontal. A clipping path, dump transparency key (see the description of the Dump -> All -> Raster option below) and sequence number may also be specified. After the popup window is exited, the user must fix the location of the text object within the composite image by moving the mouse cursor to the appropriate position and then clicking any mouse button (unless the text is being rendered in path mode, in which case the location is meaningless). This location will vertically correspond to the baseline of the first line of text. For flush left and justified text, it will horizontally correspond to the left edge of the text. For flush right text, it will horizontally correspond to the right edge of the text. For centered text, it will horizontally correspond to the center of the text.

Insert -> Vector. This option will create an internal vector object which may be used to annotate the composite image. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference via ice's menu system. The Width of the vector may be specified in points. (A width of 0 will cause the vector to be exactly one pixel wide at any resolution.) The Line Style may be either Solid or Dashed. If the vector is to be rendered in dashed mode, the Dash Style may be set to either Simple or Complex. If simple dashes are desired, the Dash Length and Gap Length may be specified in points. If complex dashes are desired, the Dash Pattern and Dash Offset may be specified in points. (The dash pattern and offset are used to construct the array and offset operands of the PostScript setdash operator. See the PostScript Language Reference Manual for a description of these parameters.) The Cap Style of the vector may be set to Butt, Round or Square. One or more triangular-shaped pointers may be affixed to either or both ends of the vector by setting the Pointers attribute appropriately. The tip of the pointer is located at the end of the vector, while its other two vertices are located at points which are equidistantly offset to either side of a point which lies further back on the vector away from the pointer tip. The exact location of these two vertices is determined by the Pointer Width, which specifies the

distance between them, and the Outside Length, which specifies the distance between the tip and the point on the vector equidistant from the vertices. If the Pointer Style is Open, the pointer will be rendered by drawing a line from each vertex to the tip. If it is Closed, an additional vertex is created on the vector at a location whose distance from the tip is specified by the Inside Length. All four vertices are then connected and the resulting polygon is filled, resulting in a solid arrowhead shape. An arbitrary foreground color, clipping path, dump transparency key (see the description of the Dump -> All -> Raster option below) and sequence number may also be specified. After the popup window is exited, the user must fix the location of the origin endpoint and terminus endpoint of the vector within the composite image by clicking the mouse at the appropriate position for each endpoint.

Insert -> Curve. This option will create an internal curve object (a Bézier curve of the type created by the PostScript curveto operator) which may be used to annotate the composite image. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference via ice's menu system. The Width of the curve may be specified in points. (A width of 0 will cause the curve to be exactly one pixel wide at any resolution.) The Line Style may be either Solid or Dashed. If the curve is to be rendered in dashed mode, the Dash Style may be set to either Simple or Complex. If simple dashes are desired, the Dash Length and Gap Length may be specified in points. If complex dashes are desired, the Dash Pattern and Dash Offset may be specified in points. (The dash pattern and offset are used to construct the array and offset operands of the PostScript setdash operator. See the PostScript Language Reference Manual for a description of these parameters.) The Cap Style of the curve may be set to Butt, Round or Square. An arbitrary foreground color, clipping path, dump transparency key (see the description of the Dump -> All -> Raster option below) and sequence number may also be specified. After the popup window is exited, the user must fix the location of the origin endpoint, the two control points, and the terminus endpoint of the curve within the composite image by clicking the mouse at the appropriate position for each of the four points.

Insert -> Marker. This option will create an internal marker object, i.e., a geometric symbol, which may be used to annotate the composite image. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference via ice's menu system. The Type of the marker may be set to Square, Triangle, Circle or

Cross, and indicates the geometric shape which the marker will take. The Size of the marker may be set to specify the radius of the object in points. Each marker has a Boundary around its perimeter and an interior Fill, either of which may be set to Opaque or Transparent. (Interior fill does not apply to cross markers.) If the marker boundary is opaque, it may be drawn with any arbitrary Boundary Width and Boundary Color. If the interior fill of the marker is opaque, it may be drawn with any arbitrary Fill Color. Horizontal and vertical scaling, rotation, clipping path, dump transparency key (see the description of the Dump -> All -> Raster option below) and sequence number may also be specified. After the popup window is exited, the user must fix the location of the center of the marker within the composite image by clicking the mouse at the appropriate position.

Insert -> Rectangle. This option will create an internal rectangle object which may be used to annotate the composite image. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference via ice's menu system. Each rectangle has a Boundary around its perimeter and an interior Fill, either of which may be set to Opaque or Transparent. If the rectangle boundary is opaque, it may be drawn with any arbitrary Boundary Width, Line Style and Boundary Color. The Line Style may be either Solid or Dashed. If the boundary is to be rendered in dashed mode, the Dash Style may be set to either Simple or Complex. If simple dashes are desired, the Dash Length and Gap Length may be specified in points. If complex dashes are desired, the Dash Pattern and Dash Offset may be specified in points. (The dash pattern and offset are used to construct the array and offset operands of the PostScript setdash operator. See the PostScript Language Reference Manual for a description of these parameters.) If the interior fill of the rectangle is opaque, it may be drawn with any arbitrary Fill Color. The rectangle's initial Dimensioning Mode may be set to either Cursor or Text. If cursor mode is selected, then the size and rotation of the rectangle are set via the cursor just after the rectangle is positioned as described below. If text mode is selected, then the Width and Height in inches of the rectangle may be set to any non-negative values, and the Rotation in degrees may be entered. Clipping path, dump transparency key (see the description of the Dump -> All -> Raster option below) and sequence number may also be specified. After the popup window is exited, the user must fix the location of the origin corner of the rectangle within the composite image by clicking the mouse at the appropriate position. The default rotation of 0 will cause the origin corner to be at the lower left of the rectangle. If cursor dimensioning mode has been selected, a second mouse click will determine an orthogonal

coordinate system about the origin corner at an arbitrary rotation, and a third mouse click will fix the location of the rectangle corner opposite the origin corner. The width and height of the rectangle will then be derived from the rotation and the two user-specified corners.

Insert -> Polygon. This option will create an internal polygon object, i.e., a set of arbitrary edge-connected vertices, which may be used to annotate the composite image. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference via ice's menu system. The Type of the polygon may be set to Closed or Open, and indicates whether or not the last vertex of the polygon should be connected to the first vertex. Each polygon has a Boundary around its perimeter and, if it is a closed polygon, an interior Fill, either of which may be set to Opaque or Transparent. If the polygon boundary is opaque, it may be drawn with any arbitrary Boundary Width and Boundary Color. If the interior fill of the polygon is opaque, it may be drawn with any arbitrary Fill Color. Horizontal and vertical scaling, rotation, clipping path, dump transparency key (see the description of the Dump -> All -> Raster option below) and sequence number may also be specified. After the popup window is exited, the user must fix the location of the first and subsequent vertices of the polygon within the composite image by clicking the mouse at the appropriate positions. The polygon must have at least three vertices. Fixing a vertex location by clicking the right mouse button indicates that the selected vertex will be the final vertex of the polygon.

Insert -> Axis. This option will create an internal axis object which may be used to annotate the composite image. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference via ice's menu system. The Origin and Terminus values of the axis may be specified, as well as the Type, which may be either Linear or Logarithmic. The terminus value must be greater than the origin value, and both values must be greater than 0 if the axis is logarithmic. The axis will be displayed as a straight line from the origin to the terminus with primary, secondary and tertiary ticks drawn perpendicularly to the line, and with a numerical annotation displayed opposite each primary tick indicating the value at that point as interpolated between the declared origin and terminus values. A primary tick will be drawn at the origin and terminus points and at regular intervals between those points. The number of secondary and/or tertiary ticks to be shown between each primary tick may be specified by declaring the number of Subdivisions. Each subdivision will be

marked by a tertiary tick, excepting the point midway between primary ticks (when the number of subdivisions is even), which will be marked by a secondary tick. The Axis Width of the primary axis line may be specified in points. (A width of 0 will cause the line to be exactly one pixel wide at any resolution.) The Tick Location may be set to Standard, Alternate or None. Standard tick location will cause the ticks to be drawn beneath a horizontal axis whose origin is to the left. Alternate tick location will cause the ticks to be drawn above such an axis, and no tick location will inhibit tick display. The height in points of primary, secondary and tertiary ticks may be independently specified, with negative values causing the ticks to be drawn on the opposite side of the axis. The Tick Width and Axis/Tick Color may also be set. Label Location, which may be set to Standard, Alternate or None, declares whether or not numeric labels should be displayed at each primary tick mark and on which side of the axis they should be drawn. The font and font size used to display the labels may be set via the Label Font and Label Font Size items. The Label Orientation may be used to declare the angle and direction with respect to the primary axis line at which the label will be displayed. The Label Offset indicates the distance in points between the label and the tick (or primary axis line, if the tick and label are being drawn on opposite sides of the axis line). The Label Color declares the color to be used to draw the labels. A clipping path, dump transparency key (see the description of the Dump -> All -> Raster option below) and sequence number may also be specified. After the popup window is exited, the user must fix the location of the origin endpoint and terminus endpoint of the axis within the composite image by clicking the mouse at the appropriate position for each endpoint.

Insert -> Path. This option will create an internal path object (consisting of a sequence of vertices connected by line segments) which may be used to control the rendering of other graphical objects. A temporary popup window is displayed which allows the specification of numerous attributes of the object. The Name of the object may be declared for purposes of future reference by other objects and by ice's menu system. The Source of the path may be set to either Interactive or File. If the former is chosen, the user must select the vertices of the path interactively by pointing and clicking the mouse after the popup window is exited. If File input is selected, XY coordinate pairs will be read from the ASCII text file specified by the Filename option, one pair per each line of the file. The coordinates will be interpreted as pixels, points, inches or user coordinates depending on the current program display units setting. The Type of the path may be specified as either Open or Closed. If the path is closed, its terminating vertex is

connected to its initial vertex. While a path object is not a renderable entity in and of itself, its location within the composite image can be shown by setting the Display option to Yes. This will cause the segments comprising the path to be drawn in the page window (but not in any raster dump of the composite image). After the popup window is exited, if the source of the path has been defined to be interactive the user must fix the location of the first and subsequent vertices of the path within the composite image by clicking the mouse at the appropriate positions. The path must have at least two vertices. Fixing a vertex location by clicking the right mouse button indicates that the selected vertex will be the final vertex of the path.

Insert -> ICE. This option will allow the user to read in a file containing ICE directives which has been previously saved with one of the program's Dump menu options. An ICE file will generally include a full description of the state of the program's page and default attributes and all or part of the graphical object and path lists at the time of the file's creation. A temporary popup panel provides for the specification of a filename and allows the user to either continue to work within the current page and default attributes or to replace those attributes with the attributes contained within the file. All graphical objects and paths described by the file will be added to the existing object and path lists. If the current default attributes are to be used, the user can decide whether attributes of new objects described by the file which have been set to a global value will (i) be deglobalized and preserve the values at which they were originally stored into the file, or (ii) reference the global value described by the current default attributes. If the new default attributes are to be used, the user can decide whether attributes of existing objects which have been set to a global value will (i) be deglobalized and preserve their current value, or (ii) reference the global value described by the new default attributes.

Delete. This option invokes a pullright menu which allows the user to delete various types of graphical objects from the composite image.

Delete -> Select. This option will delete the graphical object selected by the next mouse click from the composite image. If the selected object is a composite object, all child objects contained within it will also be deleted.

Delete -> PS Document -> [filename]. This option will delete the named external PostScript file from the composite image.

Delete -> Raster -> [filename]. This option will delete the named external Sun rasterfile from the composite image.

Delete -> Text -> [objectname]. This option will delete the named text object from the composite image.

Delete -> Vector -> [objectname]. This option will delete the named vector object from the composite image.

Delete -> Curve -> [objectname]. This option will delete the named curve object from the composite image.

Delete -> Marker -> [objectname]. This option will delete the named marker object from the composite image.

Delete -> Rectangle -> [objectname]. This option will delete the named rectangle object from the composite image.

Delete -> Polygon -> [objectname]. This option will delete the named polygon object from the composite image.

Delete -> Axis -> [objectname]. This option will delete the named axis object from the composite image.

Delete -> Composite -> [objectname]. This option will delete the named composite object and all child objects contained within it from the composite image.

Delete -> Path -> [objectname]. This option will delete the named path object from the composite image. The deletion of a path which is currently referenced by another object will not be permitted.

Delete -> All. This option will delete all objects (including paths) from the composite image.

Attributes. This option invokes a pullright menu which allows the user to change various attributes of the composite image or its component graphical objects.

Attributes -> Page. Change overall composite image characteristics and control settings via a temporary popup panel which is displayed when this menu option is invoked. The dimensions and resolution of the composite image in inches and dots per inch, respectively, may be specified here. Update Mode can be toggled between Automatic, which will cause the on-screen image to be regenerated whenever objects are loaded, unloaded, modified, etc., and Manual, which inhibits all image regeneration except when specifically requested by the user via the Redisplay menu option. The Location Mode controls the manner in which the user specifies page coordinates for various operations such as object

insertion and translation. It may be set to Cursor, which causes coordinate specification to be performed by positioning the mouse cursor to the desired location and clicking a mouse button, or Text, which causes such specification to be done by typing the desired coordinates (in terms of the current display units) into a popup panel. If cursor mode is selected, operations which involve positioning an object cause a set of crosshairs to track the location of the mouse cursor within the page window, and display the current cursor coordinates immediately adjacent to the center of the crosshairs. This coordinate display, which may obscure underlying objects in the page window, can be inhibited by setting the Location Display appropriately. The Display Units item, which may be set to Pixels, Points, Inches or User Defined, determines the format of the (x, y) coordinate values that are displayed or requested whenever such operations involving image location are performed. If User Defined is selected, the user may specify independent horizontal and vertical scales in user units per inch, as well as a mapping between inches and user units for a single reference point within the image. The Clipping Path item may be used to specify a global clipping path. The rendering of all PostScript objects, both external and internal, will be confined to the interior of this path. (Raster objects are not clipped.) Note that individual objects may be further clipped to whatever clipping path has been specified in their particular attribute panels. Object Origin Highlight, when enabled, will cause a small square marker to be drawn at the origin location of each object. This may be useful when attempting to select a particular object with the mouse from a group of closely spaced objects. A Background Color may be specified (default white), which is used to initialize the composite image before any object rendering is performed.

Attributes -> Default. Change various default attributes for component graphical objects via a temporary popup panel which is displayed when this menu option is invoked. These attributes include font and font size (for text and axis objects), line width (for vector, curve and axis objects), foreground and background colors (for monochrome rasters and text, vector, curve and axis objects), marker type and radius (for marker objects), boundary width, boundary color and fill color (for marker and polygon objects) and dump transparency key (for external PostScript documents and all internal objects). A default attribute value will be used by any object whose particular value for that attribute has been set to Default.

Attributes -> Insert. This option may be used to affect the initialization of attributes for new objects created by the Insert -> [object-type] menu options described above. When

this option is selected, a temporary popup panel is displayed which allows the user to declare whether New Object Attributes should be initialized to Default values or to the values specified in the Last Edit of any object of the same type (made via either an Insert -> [object-type] or Attributes -> [object-type] edit).

Attributes -> Select. This option may be used to modify attributes of the atomic graphical object selected by the next mouse click.

Attributes -> PS Document -> [filename]. This option may be used to modify the scaling, rotation, dump transparency key or sequence number of any external PostScript file which has been previously loaded.

Attributes -> Raster -> [filename]. This option may be used to modify the pixel replication, orientation, display mode or sequence number of any external Sun rasterfile which has been previously loaded. If the rasterfile is a 1-bit file, the foreground and background colors may also be modified. (Due to certain window system resource limitations and implementation details, certain combinations of foreground and background values may result in the respective on-screen rendering of foreground and background as black and white.)

Attributes -> Text -> [objectname]. This option may be used to modify the various attributes of a previously created text object such as font, size, scaling, color, the text itself, etc.

Attributes -> Vector -> [objectname]. This option may be used to modify the various attributes of a previously created vector object such as width, line style, dash style, cap style, color, etc. The origin and terminus endpoints of the vector may be moved by clicking the appropriate command button in the displayed popup panel and then clicking the mouse at the desired position within the image.

Attributes -> Curve -> [objectname]. This option may be used to modify the various attributes of a previously created curve object such as width, line style, dash style, cap style, color, etc. The origin and terminus endpoints and the two control points of the curve may be moved by clicking the appropriate command button in the displayed popup panel and then clicking the mouse at the desired position within the image.

Attributes -> Marker -> [objectname]. This option may be used to modify the various attributes of a previously created marker object such as geometric type, size, boundary and fill characteristics, scaling, rotation, etc.

Attributes -> Rectangle -> [objectname]. This option may be used to modify the various attributes of a previously created rectangle object such as boundary and fill characteristics, rotation, etc.

Attributes -> Polygon -> [objectname]. This option may be used to modify the various attributes of a previously created polygon object such as boundary and fill characteristics, scaling, rotation, etc.

Attributes -> Axis -> [objectname]. This option may be used to modify the various attributes of a previously created axis object such as origin and terminus values, tick and label characteristics, etc. The origin and terminus endpoints of the axis may be moved by clicking the appropriate command button in the displayed popup panel and then clicking the mouse at the desired position within the image.

Attributes -> Path -> [objectname]. This option may be used to modify the closure and display attributes of a previously created path object.

Translate. This option invokes a pullright menu which allows the user to change the location of component graphical objects within the composite image. The manner in which this is done is dependent upon the global location mode. If cursor location is enabled, an object is repositioned by moving the mouse cursor to the appropriate location and clicking any mouse button. If text location is enabled, the user will be prompted to enter the new object origin coordinates via the keyboard into a temporary popup window.

Translate -> Select. Change the location of the graphical object selected by the next mouse click. After selecting the object, the user may reposition it as described above. If the selected object is a composite object, the reference location displayed at the center of the crosshairs in the page window (or in the location popup window) will be that of the atomic member of that composite whose origin is closest to the point at which the mouse was clicked, and all atomic objects contained within the composite will be translated relative to the reference location.

Translate -> PS Document -> [filename]. Change the location of an external PostScript file within the image. After selecting this option, the user may reposition the object.

Translate -> Raster -> [file name]. Change the location of a raster within the image. After selecting this option, the user may reposition the object.

Translate -> Text -> [objectname]. Change the location of a text object within the image. After selecting this option, the user may reposition the object.

Translate -> Vector -> [objectname]. Change the location of a vector object within the image. After selecting this option, the user may reposition the object.

Translate -> Curve -> [objectname]. Change the location of a curve object within the image. After selecting this option, the user may reposition the object.

Translate -> Marker -> [objectname]. Change the location of a marker object within the image. After selecting this option, the user may reposition the object.

Translate -> Rectangle -> [objectname]. Change the location of a rectangle object within the image. After selecting this option, the user may reposition the object.

Translate -> Polygon -> [objectname]. Change the location of a polygon object within the image. After selecting this option, the user may reposition the object.

Translate -> Axis -> [objectname]. Change the location of an axis object within the image. After selecting this option, the user may reposition the object.

Translate -> Composite -> [objectname]. Change the location of a composite object within the image. After selecting this option, the user may reposition the object. The reference location displayed at the center of the crosshairs in the page window (or in the location popup window) will be that of the first atomic object inserted into that composite. All atomic objects contained within the composite will be translated relative to the reference location.

Translate -> Path -> [objectname]. Change the location of a path object within the image. After selecting this option, the user may reposition the object.

Copy. This option invokes a pullright menu which allows the user to create a copy of an existing graphical object. The user will be prompted for the name of the new object. (An arbitrary name will be selected if no name is specified.) The new object must then be positioned in the same manner employed by the Insert and Translate options as described above.

Copy -> Select. This option will copy the graphical object selected by the next mouse click. If the selected object is a composite object, all child objects contained within it

will also be copied.

Copy -> PS Document -> [filename]. This option will copy the object corresponding to the named external PostScript file.

Copy -> Raster -> [filename]. This option will copy the object corresponding to the named external Sun rasterfile. Note that unlike the raster insertion process, the raster will be fully loaded before the location procedure is initiated, so there may be a significant delay before the positioning bounding box is displayed.

Copy -> Text -> [objectname]. This option will copy the named text object.

Copy -> Vector -> [objectname]. This option will copy the named vector object.

Copy -> Curve -> [objectname]. This option will copy the named curve object.

Copy -> Marker -> [objectname]. This option will copy the named marker object.

Copy -> Rectangle -> [objectname]. This option will copy the named rectangle object.

Copy -> Polygon -> [objectname]. This option will copy the named polygon object.

Copy -> Axis -> [objectname]. This option will copy the named axis object.

Copy -> Composite -> [objectname]. This option will copy the named composite object and all child objects contained within it.

Composite. This option invokes a pullright menu which provides access to several types of operations affecting composite objects.

Composite -> Bind. Bind a collection of existing atomic and/or composite objects into a new composite object. A popup panel allows the specification of the name of the new composite. After the popup panel is exited, the existing objects which are to be bound together may be selected either by clicking the left or middle mouse button while the cursor is positioned over them, or by using the right mouse button to invoke a menu which allows the selection of objects by name. The selection process may be terminated either by selecting the last object to be included with the

left or middle mouse button while either the <SHIFT> or <CONTROL> key is depressed, or by selecting the Done option from the menu that is invoked by the right mouse button.

Composite -> Attributes. This option invokes a pullright menu that allows the user to change the scaling and rotation of an existing composite object as a whole via a temporary popup panel. The Scale of the object may be changed by setting it to any non-zero positive number. This will cause all atomic objects belonging to the affected composite to be resized and repositioned, using the page origin at the lower left corner as a reference point. Note that scaling will not affect the internal attributes of child objects (such as font size, line width, etc.) unless the Internal Scale option is set to Yes and the internal attributes do not reference a global value. An arbitrary counterclockwise Rotation about the page origin may also be specified.

Composite -> Attributes -> Select. Change the scaling and rotation of the composite object selected by the next mouse click.

Composite -> Attributes -> [objectname]. Change the scaling and rotation of the named composite object.

Composite -> Unbind. This option invokes a pullright menu that allows the user to unbind an existing composite object and remove all association between its child objects.

Composite -> Unbind -> Select. Unbind the composite object selected by the next mouse click.

Composite -> Unbind -> [objectname]. Unbind the named composite object.

Composite -> Add. This option invokes a pullright menu that allows the user to add one or more new atomic or composite children to an existing composite object. See the Composite -> Bind option above for a description of how the new child objects may be selected.

Composite -> Add -> Select. Add one or more new atomic or composite children to the existing composite object selected by the next mouse click.

Composite -> Add -> [objectname]. Add one or more new atomic or composite children to the existing named composite object.

Composite -> Remove. This option invokes a pullright menu that allows the user to remove one or more child objects from an existing composite object. The association between

the child object and the composite object is destroyed, but the child object itself is not deleted. See the Composite -> Bind option above for a description of how the child objects to be removed may be selected.

Composite -> Remove -> Select. Remove one or more child objects from the existing composite object selected by the next mouse click.

Composite -> Remove -> [objectname]. Remove one or more child objects from the existing named composite object.

Colormap. The program uses one colormap for all PostScript images and 1-bit rasterfiles, and one colormap for each color rasterfile. Only one colormap may be active at any one time, meaning that many of the objects will be displayed with false colors. This option invokes a pullright menu which allows the user to activate either the default colormap or the colormap associated with any particular color rasterfile which has been previously loaded.

Colormap -> Default. Activate the default colormap to correctly view all PostScript objects and 1-bit rasterfiles.

Colormap -> [filename]. Activate the colormap associated with a particular color rasterfile to correctly view that raster.

Dump. This option invokes a pullright menu which allows the user to dump either the entirety or an arbitrary portion of the composite image to a disk file in a variety of formats. An ICE format file will generally contain a full description of the current page and default attributes and a hierarchical description of the appropriate atomic, composite and path objects. Such a file can be read back into the program at any time (via the Insert -> ICE menu option) to provide a full or partial reconstruction of the present program environment, object list and composite image. A PostScript format file will consist of fully self-contained PostScript descriptions of the appropriate graphical and path objects (except rasters) suitable for output on any standard PostScript display device. Note that (i) ICE directives are written in the form of PostScript comments, and (ii) all input lines which are not ICE directives are ignored by the Insert -> ICE menu option, so that it is possible to write a file containing both ICE directives and PostScript output and have it be both re-readable by ice and intelligible to any PostScript output device. (PostScript-only output generated by this option can of course be read back into ice as an opaque external PostScript document, but all distinctions between separate graphical objects will be irretrievably lost.) A raster format file will be a Sun rasterfile version

of the currently displayed composite image.

Dump -> All. This option invokes a pullright menu which allows the user to dump the entire composite image to a disk file.

Dump -> All -> ICE/PostScript. Dump the composite image to a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> All -> Raster. Dump the composite image to a Sun rasterfile. A temporary popup window prompts for the name of the file to be created. If the composite image contains no raster objects, the output file will be of the same depth as the display on which the program is being run, i.e., 1 bit deep for monochrome displays and 8 bits deep for color displays. If the image does contain a raster, the output file will be made 32 bits deep to avoid potential conflicts between raster object colormaps as well as the colormap used by the server to perform all PostScript rendering. Note that while raster objects are always dumped in full color, PostScript objects can only be rendered in monochrome by the NeWS server on a monochrome display, and hence will be dumped in monochrome as well as previewed in monochrome! The user should also be aware that image dumps are very time-consuming, particularly when both PostScript and raster objects are involved. This problem can be minimized by using identical or adjacent sequence numbers and identical dump transparency keys for as many PostScript objects as possible. The following discussion, which applies only to the creation of 32-bit output files, supplies further information on the process of dumping a composite image which contains both PostScript and raster objects, and explains how dump transparency keying affects both the appearance of the final output raster as well as the efficiency with which it can be generated. As mentioned above, ice maintains an ordered internal list of the graphical objects which make up the composite image. This list, which determines the order in which the objects are rendered, is ordered primarily by object sequence number (from low to high). Within each sublist made up of objects with identical sequence numbers, there is a secondary ordering by dump transparency key. (Raster objects have no such key and are placed at the end of each sublist.) For PostScript objects, the dump transparency key represents a color which is considered to be transparent when a raster dump is performed, and which allows any underlying color from either the page background or a previously rendered object to pass through. A 32-bit raster dump is done as follows. (1) Dump memory for the output raster is created and initialized to the background color defined in the page attributes window. (2) The object

list is traversed from low sequence number to high sequence number. (2-a) When a PostScript object is encountered, (2-a.1) a maximal set of PostScript objects contiguous within the object list with dump transparency keys identical to that of the encountered object is constructed, (2-a.2) a scratch image memory is initialized to the dump transparency key of the objects in the set, (2-a.3) all objects in the set are drawn into the scratch image memory in the order in which they appear on the object list, and (2-a.4) each pixel in the scratch image memory which is not set to the dump transparency key is copied into the dump memory. (2-b) When a raster object is encountered, it is drawn directly into the dump memory at the highest possible bit resolution. (For instance, if the original rasterfile is a 24-bit file, the original 24-bit colors will be dumped, not the approximated 8-bit colors which are used to render the on-screen image on an 8-bit color display. If the file is a 1-bit file, the 24-bit foreground and background colors specified in the raster attributes panel will be used in the output image. Even on a monochrome display, all raster objects are dumped in full color.) Finally, (3) the dump memory is copied to the output file and then freed.

Dump -> Select. Dump the object selected by the next mouse click into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents. If the selected object is a composite object, all of its child objects will be dumped as well.

Dump -> PS Document -> [objectname]. Dump the named external PostScript object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Raster -> [objectname]. Dump the named external raster object into a file in ICE format. A temporary popup window prompts for the name of the file to be created. (ice does not currently dump raster objects in PostScript format.)

Dump -> Text -> [objectname]. Dump the named text object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Vector -> [objectname]. Dump the named vector object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Curve -> [objectname]. Dump the named curve object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Marker -> [objectname]. Dump the named marker object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Rectangle -> [objectname]. Dump the named rectangle object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Polygon -> [objectname]. Dump the named polygon object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Axis -> [objectname]. Dump the named axis object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Composite -> [objectname]. Dump the named composite object and all of its children into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents.

Dump -> Path -> [objectname]. Dump the named path object into a file in ICE and/or PostScript format. A temporary popup window prompts for the name of the file to be created and the format of its contents. Note that the current page and default attributes are not dumped by this option as they have no effect upon paths.

Redisplay. Force a complete regeneration of the on-screen composite image display.

Exit. Exit the program.

ENVIRONMENT

A number of environment variables may be used to control the operation of the program.

NEWSSERVER

This value is used to specify the NeWS server to which the program should connect. Its use is mandatory unless the program has been invoked with the -display option, in which case it will be ignored. In most cases it is

automatically set by the X11/NeWS server startup sequence.

OPENWINHOME
This value is used to determine the location of the OpenWindows fonts. The use of either OPENWINHOME or FONTPATH is mandatory.

FONTPATH
This value is used to override the OPENWINHOME variable with regard to the location of the OpenWindows fonts. The use of either OPENWINHOME or FONTPATH is mandatory.

ICEDOCPATH
This colon-separated list of directories will be searched any time the user requests that some external file (e.g., PostScript file or Sun rasterfile) be read. If it is not set, only the current directory will be searched. A valid example would be .:$HOME/psfiles:$HOME/rasters.

ICETMPDIR
This directory will be used to store the temporary file created by using the -ps option. If it is not set, the current directory will be used.

X DEFAULTS
The following X11 defaults may be used to control the operation of the program.

LXTInternalBorderWidth
Set the internal border width of all top-level windows. This is advisable when running the program under the OpenWindows 2.0 olwm(1) window manager on a monochrome system. (Default is 0.)

Geometry
Set the initial size of the top-level window containing the page display. (Default is 600x600.)

Font Set the font to be used for all menus and interactive panels. (Default is 8x13.)

BorderWidth
Set the border width to be used for all top-level windows. Note that this value is ignored by the OpenWindows 2.0 version of the olwm(1) window manager. (Default is 2.)

Foreground
Set the foreground color to be used for all top-level windows. (Default is black.)

Background
Set the background color to be used for all top-level windows. (Default is white.)

Border
Set the border color to be used for all top-level windows. (Default is black.)

SEE ALSO
rasterfile(5)

BUGS
The handling of text objects is rather primitive, and is intended to support simple annotations rather than large-scale typesetting. In particular, the handling of tab characters and justified text is extremely naive.

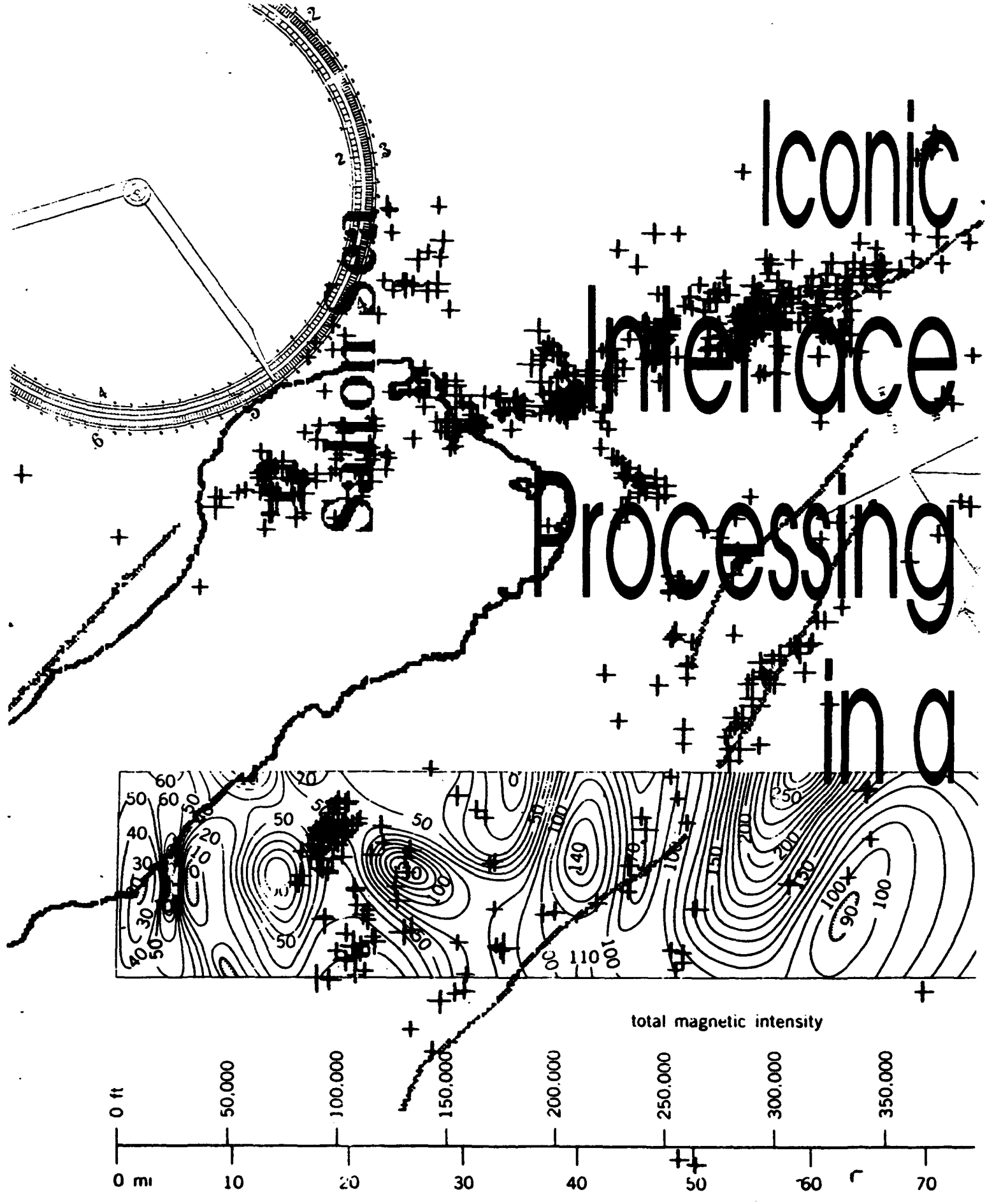
Raster objects are not included in PostScript dumps.

Since all on-screen PostScript rendering is performed by the underlying NeWS server, ice inherits any bugs present in Sun's PostScript rasterizer.

Reliance on features of both the X11 and NeWS window systems cause this program to be not easily portable to any window system other than Sun's OpenWindows.

AUTHOR
Roger Davis, June 1990.

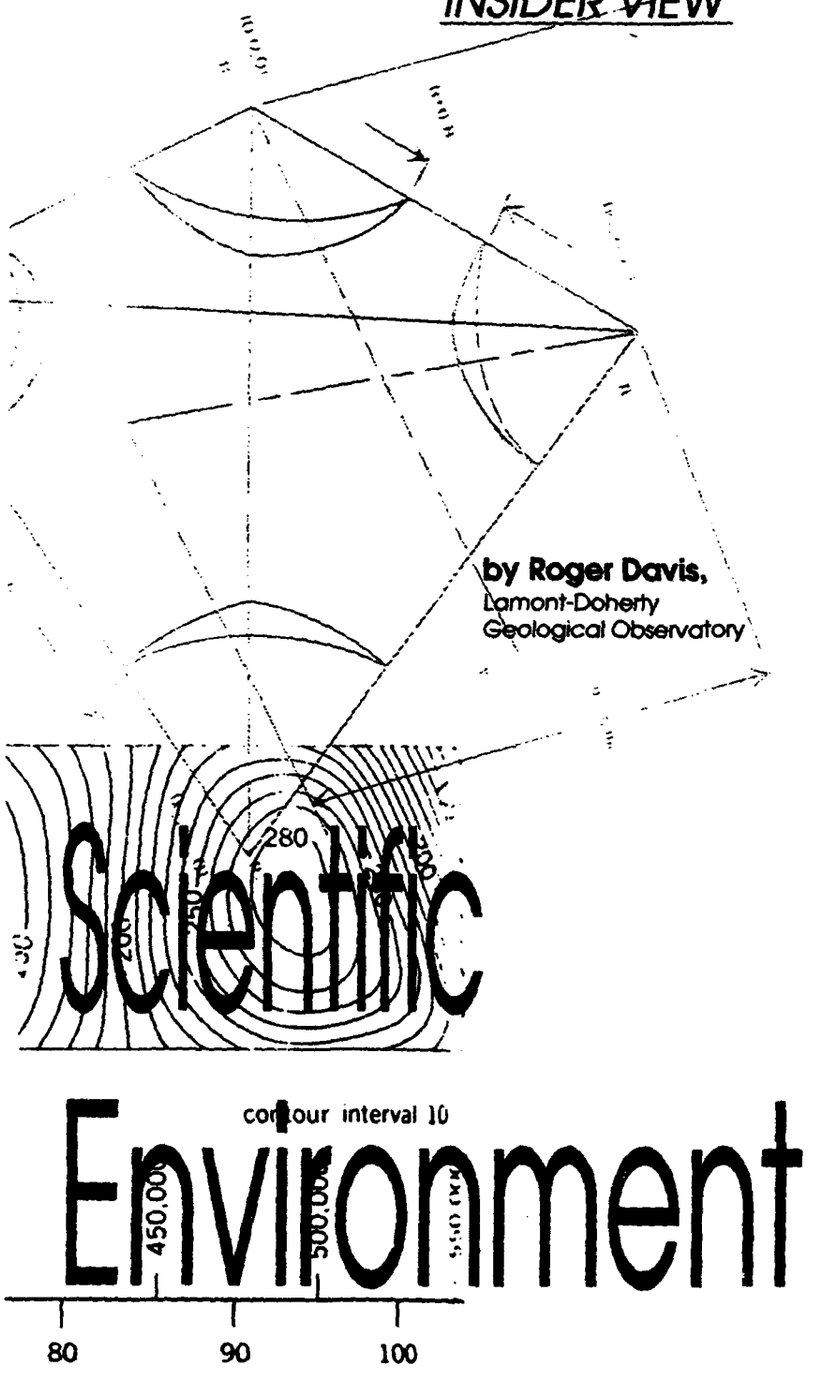
Iconic Interface Processing in a



INSIDER VIEW

Scientific Environment

by Roger Davis,
Lamont-Doherty
Geological Observatory



The Lamont-Doherty Geological Observatory, an academic institution operated by Columbia University, performs basic research in the earth sciences. Its departments encompass a variety of disciplines such as seismology, oceanography, climatology, geology, geophysics, geochemistry and marine biology. The Lamont work force includes three roughly equal-sized groups of scientists, graduate students and technical support staff.

Most research at Lamont involves large amounts of data that can be meaningfully studied only with the aid of computers. Three years ago, the Observatory changed its computing environment from a small collection of unconnected department minicomputers running a variety of operating systems to a UNIX-dominated network connecting about a dozen Sun file servers, forty-odd diskless Sun clients, a small number of Masscomp and Silicon Graphics workstations, and an assortment of Macintoshes and PCs.

The Observatory as a whole is strongly committed to Sun as a primary vendor, but has working relationships with a large number of academic and government institutions using other computing hardware. As a result, portability is a primary consideration with the design of local software and selection of outside software for our use. The Observatory stresses the use of publicly acknowledged or de facto standards such as X11, XDR, etc.

Classical Processing Techniques and Problems

One of the most common techniques for scientific analysis at Lamont involves massaging an initial collection of data sets via a group of related programs performing various functions into some final state. The processing steps will often be repeated a number of times with minor variations, either to arrive at a single "correct" answer or to construct a variety of outputs that can be usefully compared. The final result of the processing is often a visual object such as a map, image, Cartesian

plot or time series.

Over the years, a number of such software suites have been developed by the Observatory's researchers and professional programming staff to deal with different classes of scientific problems. These packages embody a programming philosophy that attempts to balance the need to find innovative ways to look at data against the necessity for timely and efficient reduction and cataloging. The design of each package is generally based upon some common data structure, I/O format and/or graphic interface. Individual modules within a package stand alone as processing entities, but are usually combined into more complex analysis schemes. New modules can be added, existing modules modified or experimental modules tested with little impact on the overall system. Examples of such packages include the AH seismic signal processing programs, the GMT PostScript mapping software and the IMG image processing tools.

While the UNIX paradigm of connecting small, single-function program modules into larger chains of processing pipelines is basically a good match for this style of analysis, a number of limitations of the command interpreters under which the processing is performed (e.g., csh and sh) have been recognized. Some of these limitations completely prevent sophisticated users from performing certain types of processing and force them to waste resources on the design and programming of software systems that are too customized to be of general use. Other limitations are more a matter of interface style, which makes it difficult for inexperienced users such as visiting scientists or new graduate students to easily master the use of some of our larger software systems. The increasing number and size of these systems have created a need for a more powerful, intuitive and standardized interface to aid in the development of new processing schemes and the construction of routine applications for more stable procedures.

One serious functional limitation of a typical interpreter is its inability to support arbitrarily complex process network topologies. The command interface of the interpreter does not allow the user to create anything more complex than a linear sequence of processes connected through stdin and stdout, such as:

```
module_1 < infile | module_2 |
module_3 > outfile
```

The tee facility often used with such interpreters is not really a true branching duplicated data stream, but merely a dead end into a file. A real tee implementation should let you do something like the scheme below.

This is of course not easy to do with a textline-based interpreter.

A related problem is the inability to connect processes via any data stream other than stdin, stdout and stderr. The only way to direct most processes to read more than one input, for example, is to specify filenames on an interpreter command line (or, even worse, in the program's code). It is impossible to connect more than one input stream of a process to the output streams of other processes.

Remote execution is another issue related to process network topology. The rsh utility can be used to stream data to and from a process running on a remote host. However, connecting several processes on different machines is difficult without turning an interpreter command line into metacharacter stew, and again, only stdin and stdout connections are supported.

A less serious but still important matter is the difficulty of teaching an unfamiliar user how to work with a given collection of processing modules. Some of the modular analysis packages mentioned above consist of thirty to fifty separate programs, each providing a separate function and utilizing an arbitrary (if hopefully consistent) UNIX command line option syntax. Even experienced users are often not aware of all of the modules that may be available in a package. It would be helpful if the interpreter could somehow be made aware of the existence of modules, as well as their command syntax, in order to make the entire processing system essentially self-documenting.

The Iconic Processing Model

These observations of the limitations in functionality and ease of use of csh and sh led directly to the development of ikp. Generally known as the icon processor, ikp is not intended as a replacement for the traditional interpreters, which are extremely powerful and flexible tools. Rather it is an adjunct utility that enables users to:

- perform certain types of processing that are otherwise impossible to achieve,
- work with an arbitrary number of software packages in a maximally intuitive manner.

Because of the importance of graphical display output to much of our analysis software as well as the importance of general portability across UNIX workstation platforms, ikp has been implemented using the X11 Window System. It makes extensive use of a locally developed Xlib-based toolkit known as LXT (the Lamont X Toolkit), which provides much of the high-level user interface functionality of Sun's proprietary SunView toolkit.

Usually started from an xterm or other terminal emulator running under X11, ikp opens a single window on the display, the *net* window. A pop-up menu system invokable from

this window provides access to all of the program's functionality. Using `ikp` is simple–the user places icons representing windows within the net window, graphically connects them into a *process net*, and then executes the net. The implementation is based upon the traditional UNIX `fork/dup/exec` model. Data is streamed directly from one process to another over a pipe or socket.

Process Creation. Through the menu system, the user can create a process, represented as an icon drawn in the net window. The name of the program to be run by the process is displayed within the icon. Small openings around the icon's perimeter known as *connectors* represent potential data streams to or from the process. There are two basic classes of processes in `ikp`, *generic* processes and *library* processes.

Generic processes are essentially abstract I/O frameworks in a variety of common configurations that can be used to run any program. A *source* process is set up to write to `stdout`. A *sink* process reads from `stdin`. A *filter* process reads from `stdin` and writes to `stdout`. A *custom* process can be configured to read from and write to any arbitrary number of file descriptors. A special kind of generic process, the *tee* process, represents the "real tee" wished for above–it duplicates an input stream read from `stdin` onto two identical output streams.

Aside from the special-purpose tee process, which is hardwired to fork the `ikp_tee` utility, generic processes have no a priori association with any particular program. The user makes this association by calling up the process edit panel bound to a particular process icon and specifying the program to be run by that process together with its command options. The user can also set the working directory and execution host of the process.

Library processes, on the other hand, are permanently associated with a particular program. At startup, `ikp` reads a site program description file that describes all of the programs in each of our major analysis packages. Each package is viewed as a library of modules, where a module is just an executable program. The description file provides information on the I/O configuration of each module, i.e., which file descriptors are used by the module to read or write, as well as complete instructions for the creation of the module's process edit panel.

A library process edit panel is considerably more complex than the edit panel of a generic process. It does not permit users to select the program to be run by the process, which is permanently fixed, but rather allows the program's command options to be set via a collection of interactive visual panel objects such as choice, cycle, toggle and type-in text items. This useful feature relieves users of the burden of remembering the command option syntax for large numbers of programs. The user does not even need to know in advance that a particular module (or library, for that matter) exists, since `ikp` constructs a menu for each library declared in the description file. The menu contains entries for all modules in the library, and is linked to the main process creation menu. If on-line documentation is available for a particular module, its edit panel will contain a help button that displays a text window providing more detailed information. Users with private software packages can create description files of their own that may be dynamically loaded by `ikp` at any time.

Users with private software packages can create description files of their own.

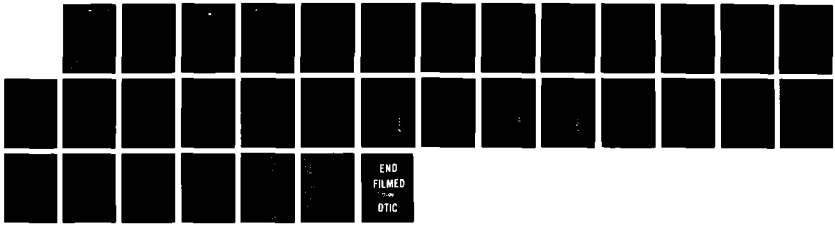
Process Net Connection. Once the user has created a number of distinct processes, they can be assembled into a process net. A small arrow is displayed adjacent to each process connector just outside the process icon perimeter to indicate the direction of data flow. The menu system allows the input and output connectors of the various processes to be joined with one another via line segments that represent the transfer of data between processes. It is also possible to bind a process connector directly to a disk file or ground it by connecting it to `/dev/null`.

Custom and library process connectors display a number just inside the icon perimeter that indicates the actual file descriptor to which the connector is mapped. Input and output connectors that do not explicitly display their underlying descriptor are mapped to either `stdin` or `stdout`. Certain output connector arrows are drawn with a dashed line to indicate that they are *diagnostic* connectors, intended primarily for the conveyance of error messages. Diagnostic connectors are usually mapped to `stderr`.

When all input and non-diagnostic output connectors within a process net have been linked to other processes, bound to files, or grounded to `/dev/null`, the net may be executed. Data from all unconnected diagnostic connectors are redirected to the `stderr` of `ikp`; `stderr` will generally be the controlling terminal for `ikp`.

Process Net Execution. A process net is executed by sequentially forking each process in the net. Process I/O is handled by opening files, pipes and/or sockets prior to the fork. The descriptors obtained thereby are duplicated into the descriptors to which the connectors are mapped in the child process between the fork and the exec of the appropriate program. A simple algorithm is used to determine the optimal order in which the processes should be forked so that `ikp` does not run out of file descriptors, which are a scarce resource on many UNIX systems. In general, forking large numbers of processes with a few connectors each will never be a problem, but forking one or more processes with a quantity of connectors approaching the system's per-process descriptor limit may present difficulties.

AD-A200 950 SATELLITE IMAGERY AND TOPOGRAPHIC DATA IN VERIFICATION 2/2
(U) LAMONT-DOHERTY EARTH OBSERVATORY PALISADES NY
D SIMPSON ET AL. 28 SEP 93 PL*-TR-93-2211
UNCLASSIFIED F19628-89-K-0007 NL



Fortunately the majority of programs in use at Lamont use only the three `stdio` descriptors, and almost none use more than five or six descriptors.

The status of each process in an executing net is continually monitored by `ikp`, and appropriate visual feedback is provided to the user. When a process commences execution, the interior color of its icon changes from white to light gray. If a process is suspended by a signal, the icon color changes to dark gray. When the process finally terminates, its icon color reverts back to the original white.

Any process exit status other than zero is deemed to be an error condition, as is any termination or suspension due to a UNIX signal. When such a condition arises, the icon of the process in question is highlighted in reverse video and the user is prompted to either abort the entire net execution or simply ignore the condition. The user can also selectively terminate or suspend individual processes within the net by invoking a menu option and clicking the mouse over the appropriate process icon.

No editing operations may be performed on any piece of an executing net, but the user is free to create or edit other nets while execution is in progress.

Remote Execution. Any process in a net may be executed on a remote machine by specifying the name of the remote host in the process edit panel. Remote execution is typically used to distribute the computational load of a large process net across several machines or to access some special-purpose hardware device, such as a tape drive or floating point accelerator, not available on the local workstation. There are three issues relating to remote execution via `ikp` that affect the user:

- remote host availability,
- user authentication and permissions,
- remote execution environment.

A process may be remotely executed only on a host that is running the

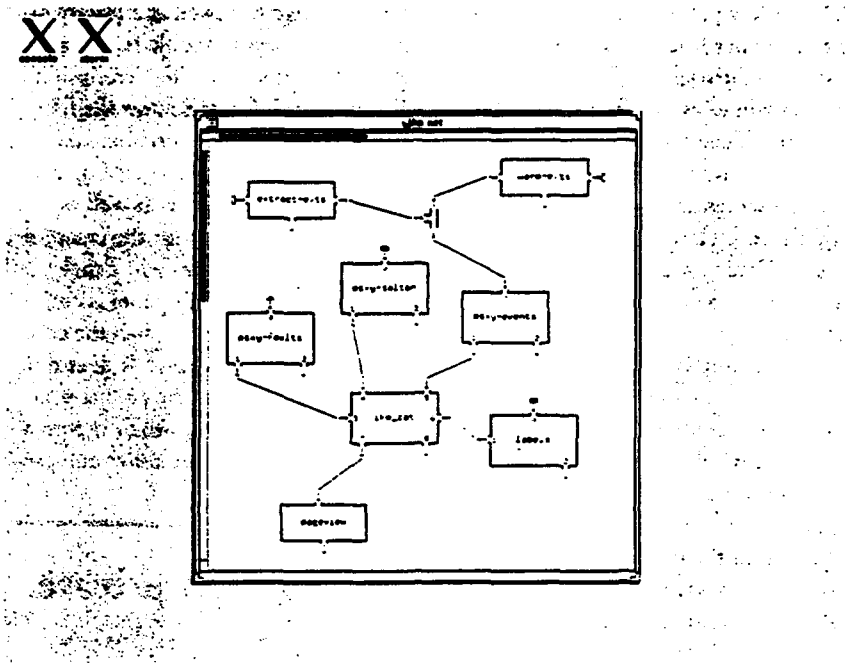


Figure 1. Sample process network, courtesy of Mark Petersen.

`ikpd` remote execution service daemon. This program, normally started up at system boot time, runs continuously as a background process. Its sole function is to service remote process execution requests from users who are running `ikp` on another machine on the network. By listening for requests at a publicly advertised address and forking a copy of itself to handle each request while the parent daemon continues listening for further requests, `ikpd` follows the standard UNIX network service daemon model.

The `ikpd` service daemon, running on the remote machine on which process execution is being requested, performs a user authentication check before granting execution privileges. This authentication check is similar to that performed by `rlogin` and `rsh`:

- a lookup of the user name in the remote password database is performed on the remote machine to determine login privileges,
- the `.rhosts` file in the user's home directory on the remote machine is scanned for the hostname of the machine from which remote execution is being requested.

If either of these steps fails, remote execution privilege is denied.

If the remote authentication check succeeds, `ikp` (and the cooperating remote `ikpd`) will attempt to construct a remote execution environment before actually running the remote process. Essentially, the user's environment is copied over the network, and a small set of transformations on that environment is performed to more appropriately tailor it to the remote machine. The remote machine may have a different architecture than the local machine and may therefore require a different executable search path, etc.

A number of implementation schemes for remote execution were considered and rejected. RPC was deemed unsuitable because of its procedural orientation. RPC calls are generally used to return values from a remotely executing function, and are not easily used to pass essentially infinite length streams of data back and forth. A multiplexed single socket model would have made the network coding fairly simple, but would have involved a lot of buffering overhead on both ends of the connection.

The ultimate solution was a multi-

socket model whereby a control socket is used for status requests, message passing, and signal and exit code reporting. A separate data socket is used for each process connector. Besides being the conceptually simplest of the models under consideration, this scheme also allows two connected processes executing on different remote machines to stream their data directly between one another after `ikp` mediates all socket address issues via its control socket protocol, thus minimizing both network traffic and local CPU usage.

Miscellaneous Features. Constructing a large process net can be a time-consuming job. Fortunately, a number of labor-saving features allow the user to save an entire net into a disk file, retrieve a net from a file, or interactively copy existing nets or processes. Once a completely connected net has been saved into a file, `ikp` can be used to execute that net outside the X11 environment, typically from a shell script, dumb terminal or alternate window system.

Example Process Network

Figure 1 illustrates a typical process net created by `ikp`. This net, created by seismologist Mark Petersen, uses some database access and plotting modules to generate a PostScript image showing the location of a number of seismic events in the vicinity of the Salton Sea in southern California. It also employs a warping module to save the event data set into a file using a non-standard map projection. The example is shown running in Sun's OpenWindows environment.

The `extract-evts` process at the upper left corner of the net searches a seismic event database and extracts the small set of events that are of interest. This event set is duplicated with a tee process. One copy of the event set is modified to conform to an arbitrary map projection using the `warp-evts` process and then saved to a disk file. The second copy of the event set is passed to the `psxy-events` process, which uses the location and magnitudes of the events to create a

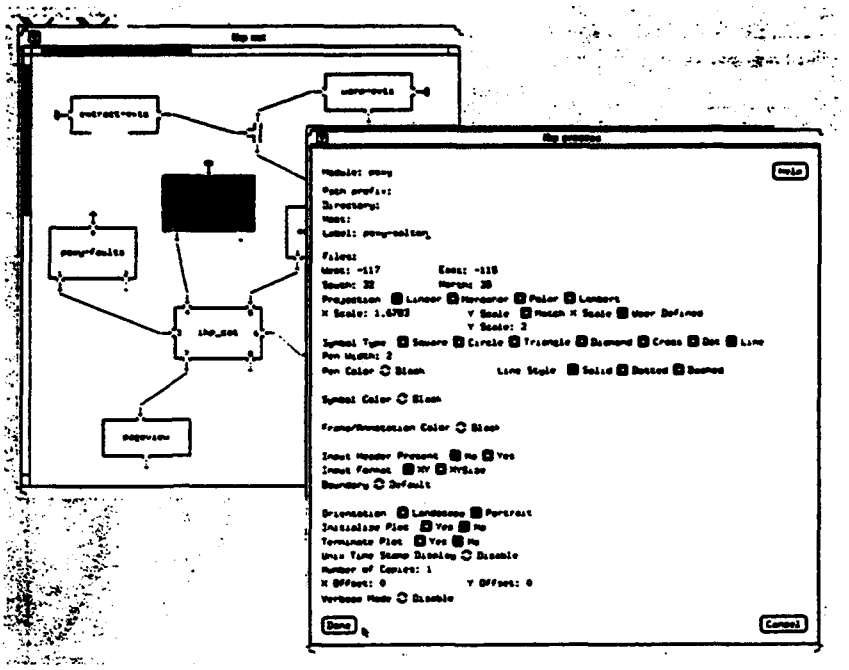


Figure 2. Process edit panel.

PostScript image consisting of a number of small graphic symbols. Each symbol represents a particular earthquake contained in the event set. The `ikp_cat` process concatenates this image with three other PostScript images created by the `psxy-faults`, `psxy-salton` and `labels` processes, which respectively display a number of seismic fault lines, the topographic outline of the Salton Sea, and a few plot labels. The merged PostScript image is then displayed with `pageview`, a PostScript previewer distributed by Sun as part of OpenWindows.

Most of the process connectors in this net are connected to other process connectors. Some are bound to disk files, which are represented by tiny rectangular file icons attached to the connector arrows. Connectors bound to `/dev/null` are marked by a ground icon identical to that used in electronic schematics. All of the diagnostic connectors in this net, drawn with dashed arrows, are unconnected, meaning that any diagnostic message written by any of the processes will be redirected to the controlling terminal of `ikp`. (In this example, the controlling terminal is actually an `xterm` that has been iconified at the upper left corner of the display.)

Figure 2 shows the process edit panel of the `psxy-salton` process. Note that the user-editable text items at the top of the panel allow the user to specify the working directory and execution host of the process, as well as a label to be displayed within the process icon. Differing labels may be used to identify several instances of the same module that may be present in a single net. The `psxy-faults`, `psxy-salton` and `psxy-events` processes in this net are actually separate instances of the `psxy` module, a PostScript plot generator that reads *x-y* input coordinates.

The `psxy` module is a library module, meaning that a description of its command option syntax and I/O configuration has been extracted from an `ikp` database of known local software. This description has been used to construct the items that fill most of the panel. The user may interactively alter these items and thereby generate the string of command option flags that are passed to the invoked executable. The command (a one liner) invoked by the displayed item settings is:

```
psxy -R-117/-115/32/35 -
Jx1.6783/2 -L2 -P -O -K -#1 -
X0 -Y0
```

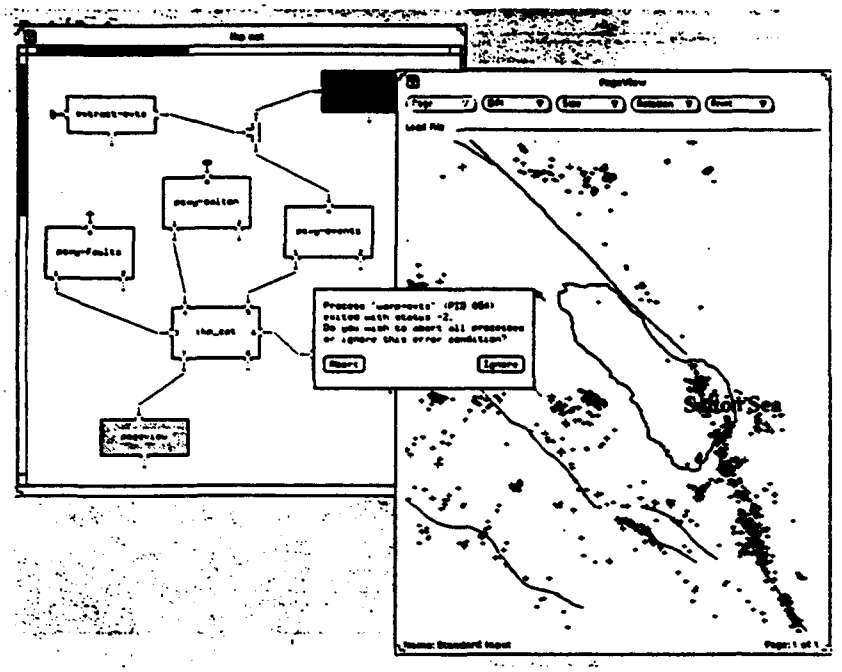


Figure 3. Executing process network.

The edit panels of ikp obviously present a friendlier interface to programs like psxy than that presented by a traditional UNIX shell interpreter.

The executing net is shown in Figure 3. The pageview previewer has processed all of its input and is displaying the concatenated image in its own window. The appearance of its icon, which is now showing a stippled gray pattern, indicates that the process is still executing. (It will continue to execute until the user either kills pageview from its own menu or tells ikp to terminate the process.) Note the appearance of the warp-evts icon at the upper right. The inverse video indicates that ikp has sensed some abnormality in its execution, which is then reported via the small pop-up window at the center of the display. In this example, certain of the command options to warp-evts were deliberately mis-set to illustrate the error handling facilities of ikp, which is able to detect abnormal exit codes or signal-induced suspension or termination of both locally and remotely executing processes.

Classical vs. Iconic Processing

The icon processor was never intended to be a complete replacement for traditional UNIX command interpreters and would be found highly inadequate as such. While traditional interpreters such as csh and its relatives are programming languages with variables, control flow, loop structures, metacharacters, etc., ikp has virtually no support for such objects and facilities. It is currently impossible, for example, to create a process net in ikp and perform an automatic sequentially looping net execution while changing input filenames at each loop iteration. (Net files written by ikp are editable ASCII text files, which could be operated on by utilities such as sed under csh control, but this would hardly be an elegant solution.) On the positive side, Lamont's icon processor has significantly expanded the ways in which users can think about processing data. It has supplied them with the ability to perform certain tasks which are outright impossible with a traditional interpreter. It has been particularly useful in performing complicated transformations on huge data sets, operations which have traditionally been difficult to accomplish on many of our systems due to a chronic lack of disk space. The software also provides the means to construct widely distributed processing systems linking remote supercomputers to local workstations. In terms of meeting its primary design goals of supporting arbitrarily complex process net topologies and significantly improving and standardizing the interface to a number of our large software systems, iconic processing is an unqualified success.

Software Availability

The ikp icon processor and all related software and documentation, as well as the LXT library package, are available free of charge via anonymous ftp from

```
lamont.ldgo.columbia.edu
```

at Internet host address 129.236.10.30. Sites without Internet access may request the software by sending a cartridge or 9-track tape and a check for $100, payable to Columbia University, to Robert Bookbinder, Lamont-Doherty Geological Observatory, Palisades, NY 10964.

Development of the icon processor was made possible by the generous support of several funding agencies including the National Science Foundation, the Defense Advanced Research Projects Agency, the U.S. Geological Survey, the Office of Naval Research, the U.S. Department of Energy and Columbia University. ➡

Roger Davis is a software engineer working in the Seismology Division of the Lamont-Doherty Geological Observatory. He is currently involved in an image processing project directed by scientist David Simpson, the chief goal of which is an increased understanding of seismological processes through the examination of satellite imagery and topographic data. The design and implementation of window system interfaces is his primary field of interest and expertise. He can be reached at rbd@lamont.ldgo.columbia.edu.

SPATIAL-DOMAIN DECONVOLUTION AND IMAGE PROCESSING

Arthur L. Lerner-Lam
Lamont-Doherty Earth Observatory
Palisades, NY

Stuart A. Sipkin
Unites States Geological Survey
Golden, CO

Filtering of digital image data is an integral part of image analysis methodologies. While a standard suite of filters is available within most image analysis software packages, new methods of analysis require new approaches. In particular, the deconvolution of landform wavelets from topographic profiles and edge detection by high-pass spatial filtering require stabilization measures due to holes in the wavenumber spectra and signal-generated noise in the raw data.

Unless the convolutional operators are short, most deconvolution and filtering methods employed in the image processing community make use of fast Fourier methods to attain computational speed advantages. This reliance on Fourier methods or short operators is conceptually simple and computationally robust, but usually is not implemented in a comprehensive or adaptive manner. Moreover, while the technical literature on the subject of filtering and deconvolutional effects is well developed, mitigation measures used in geophysical or image applications are usually ad hoc or subjective.

A typical problem is the deconvolution of zero-mean wavelets from complex data. For example, studies of mid-ocean ridge morphology have examined the utility of expressing the bathymetry as a convolution between a canonical landform shape function (a "topolet") and a response series (Malinverno, 1990). In a simple view, the response series gives information about the timing and scale of the morphological event represented by the topolet. In many cases, (such as the need to model isostatic response), the topolet has zero mean and generates a singular deconvolution filter. This destabilizes the estimation of small wavenumber components of the response function and can give unreliable estimates of long-wavelength trends.

The usual response to this destabilization is the modification of the deconvolution filter to remove the singularity. This can either be done by prior truncation of the filter below some "water level", chosen to be typically 80-100 dB below the filter maximum, or by a posteriori high-pass with similarly chosen corner. While effective, this procedure removes the opportunity to make the deconvolution adapative to either the noise environment or scientific constraints.

A spatial-domain representation of filtering and deconvolution offers the appropriate framework to make these procedures adaptive. Time-domain techniques are enjoying a resurgence in time-series processing in the seismological community, driven chiefly by the need for stabilized removal of instrument response or the analysis of boundary interaction phases (e.g. Sipkin and Lerner-Lam, 1992, Clarke and Silver, 1992). These methods have led to parametric representations of the time series for inverse problems in source characteristics and crustal structure studies (Abers et al., 1992, Sheehan et al., 1992).

We present here a formalism for developing spatial-domain deconvolution techniques which is rooted in inverse theory. This work was originally published as Sipkin and Lerner-Lam (1992), with application to instrument deconvolution.

Bulletin of the Seismological Society of America, Vol. 82, No. 1, pp. 238–258, February 1992

PULSE-SHAPE DISTORTION INTRODUCED BY BROADBAND DECONVOLUTION

BY STUART A. SIPKIN AND ARTHUR L. LERNER-LAM

ABSTRACT

The availability of broadband digitally recorded seismic data has led to an increasing number of studies using data from which the instrument transfer function has been deconvolved. In most studies, it is assumed that raw ground motion is the quantity that remains after deconvolution. After deconvolving the instrument transfer function, however, seismograms are usually high-pass filtered to remove low-frequency noise caused by very long-period signals outside the frequency band of interest or instabilities in the instrument response at low frequencies. In some cases, data must also be low-pass filtered to remove high-frequency noise from various sources. Both of these operations are usually performed using either zero-phase (acausal) or minimum-phase (causal) filters. Use of these filters can lead to either bias or increased uncertainty in the results, especially when taking integral measures of the displacement pulse. We present a deconvolution method, based on Backus–Gilbert inverse theory, that regularizes the time-domain deconvolution problem and thus mitigates any low-frequency instabilities. We apply a roughening constraint that minimizes the long-period components of the deconvolved signal along with the misfit to the data, emphasizing the higher frequencies at the expense of low frequencies. Thus, the operator acts like a high-pass filter but is controlled by a trade-off parameter that depends on the ratio of the model variance to the residual variance, rather than an *ad hoc* selection of a filter corner frequency. The resulting deconvolved signal retains a higher fidelity to the original ground motion than that obtained using a postprocess high-pass filter and eliminates much of the bias introduced by such a filter. A smoothing operator can also be introduced that effectively applies a low-pass filter. This smoothing is useful in the presence of blue noise, or if inferences about source complexity are to be made from the roughness of the deconvolved signal.

INTRODUCTION

When using broadband seismic data, it is a common practice to apply a high-pass filter with a corner frequency around 0.01 Hz after deconvolution of the instrument transfer function (Harvey and Choy, 1982; Silver and Masuda, 1985; Beck and Lay, 1986; Choy and Cormier, 1986; Silver and Chan, 1986; Vidale and Garcia-Gonzalez, 1988; Lay and Young, 1989). This is necessitated mainly by uncertainties in the transfer functions at lower frequencies. Both zero-phase (acausal) and minimum-phase (causal) filters have been used in these studies. The networks used include the World Wide Standard Seismograph Network (WWSSN), the Canadian Seismic Network (CSN), the Regional Seismic Test Network (RSTN), the Gräfenberg array (GRF), and the Global Digital Seismograph Network (GDSN) (including the Digital World Wide Standard Seismograph Network (DWWSSN)). In some cases, the intermediate-period, or broadband channel, was used, and in some cases the long- and short-period channels were combined. When combining the long- and short-period GDSN channels, it is sometimes necessary to use a high-pass filter with a corner frequency as high as 0.025 Hz (Beck and Lay, 1986). Using current state-of-the-

art broadband systems with well-calibrated transfer functions at lower frequencies, such as GEOSCOPE and IRIS, it should be possible to move the corner frequency of the high-pass filter to ~ 0.003 Hz.

Several recent studies of shear-wave propagation have noted that the broadband S waveform is often anomalously broadened relative to the ScS waveform (Beck and Lay, 1986; Choy and Cormier, 1986; Silver and Chan, 1986). In several such studies, the relative broadening was measured by comparing the centroid and/or variance, which are integral measures of the broadband displacement pulses. These integral measures can be severely biased by the filtering applied during the restoration process. Seidl and Hellweg (1988), for example, showed how estimates of the moment of a displacement pulse are affected by band-limiting a broadband input wavelet. In this paper, we quantify these biases and present a new deconvolution method that mitigates this problem.

Synthetic Tests

For our synthetic tests we use a source function of the form

$$s(\tau_1, \tau_2, t) = \exp\left[-\frac{\tau_1}{t} - \frac{t}{\tau_2}\right] H(t), \tag{1}$$

where $H(t)$ is the Heaviside function. This representation is a good approximation to the far-field displacement expected for a causally Q filtered (constant t^*) impulsive source (Futterman, 1962). The constants τ_1 and τ_2 are related to the source duration and the average dissipation along the path of propagation. For observed S and ScS waves, a representative value for τ_1 is 1 sec and τ_2 typically ranges between 3 and 6 sec. The input pulses are filtered with third-order causal and acausal Butterworth filters. The results of the following tests yield the biases expected for a best-case scenario of no noise and a perfect system transfer function.

The unfiltered source functions $s(1, 3, t)$ and $s(1, 6, t)$ and the effects of passing them through causal and acausal high-pass filters are shown in Figures 1 and 2. The filters used have corner-frequencies of 0.001, 0.003, and 0.01 Hz. It is easily seen that the pulse-shape distortion grows more severe as the corner-frequency increases. It can also be seen that the distortion is worse for the pulse with the longer duration. Silver and Masuda (1985) attempted to correct for this distortion by subtracting a linear trend between the first break and the peak of the first overshoot from the causally filtered seismograms. This procedure was subsequently used in other studies (Silver and Chan, 1986; Lay and Young, 1989) and is illustrated in Figure 3. The effect of applying this procedure to the causally filtered waveforms in Figure 1 is shown in Figure 4; both the centroid and the variance of the pulses are severely affected. Furthermore, the amount by which these quantities are shifted strongly depends on the duration of the input pulse. The centroidal parameters are listed in Table 1. For a corner frequency of 0.01 Hz, the centroid of the 3-sec pulse is shifted by 1.5 sec whereas the centroid of the 6-sec pulse is shifted by 3.9 sec. The true centroidal difference between the two pulses is 3.4 sec, but the measured difference will be 1.0 sec. The true difference in pulse duration, determined using the variance, is 29.0 sec^2 (5.4 sec), but the measured difference will be 2.5 sec^2 (1.6 sec). If the

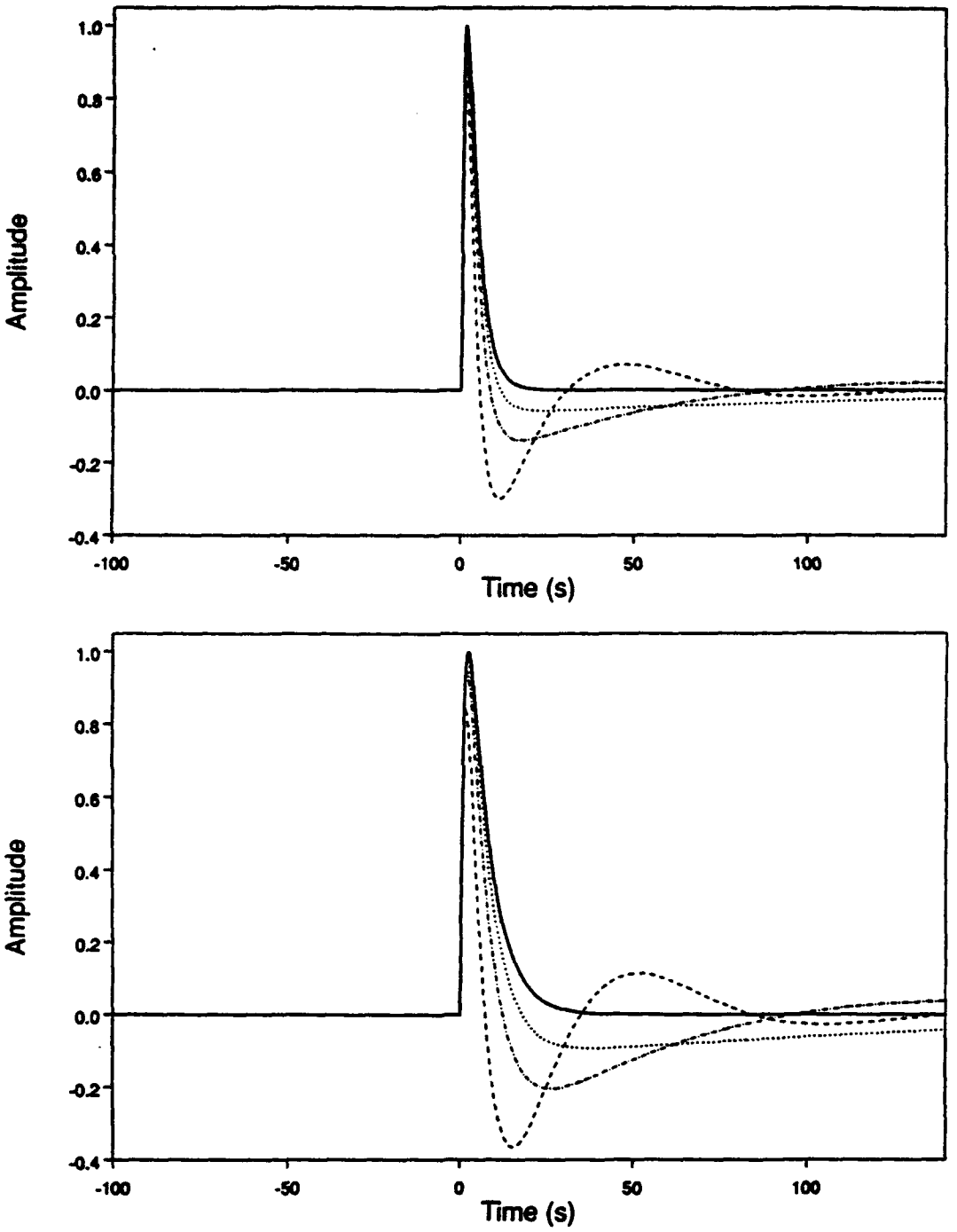


FIG. 1. Causally filtered source functions $s(1, 3, t)$ (*top*) and $s(1, 6, t)$ (*bottom*). Original source functions (heavy solid line) and source functions after applying high-pass filters with corner frequencies of 0.001 Hz (dotted), 0.003 Hz (dot-dashed), and 0.01 Hz (dashed).

pulse duration is estimated as twice the centroid time, then the difference is 6.7 sec, but the measured difference will be 2.0 sec. Not only are all of the centroidal parameters biased to low values, but their differences are biased to even lower values. This implies that, in the studies using this procedure, both the measured differences in seismological parameters and the magnitudes

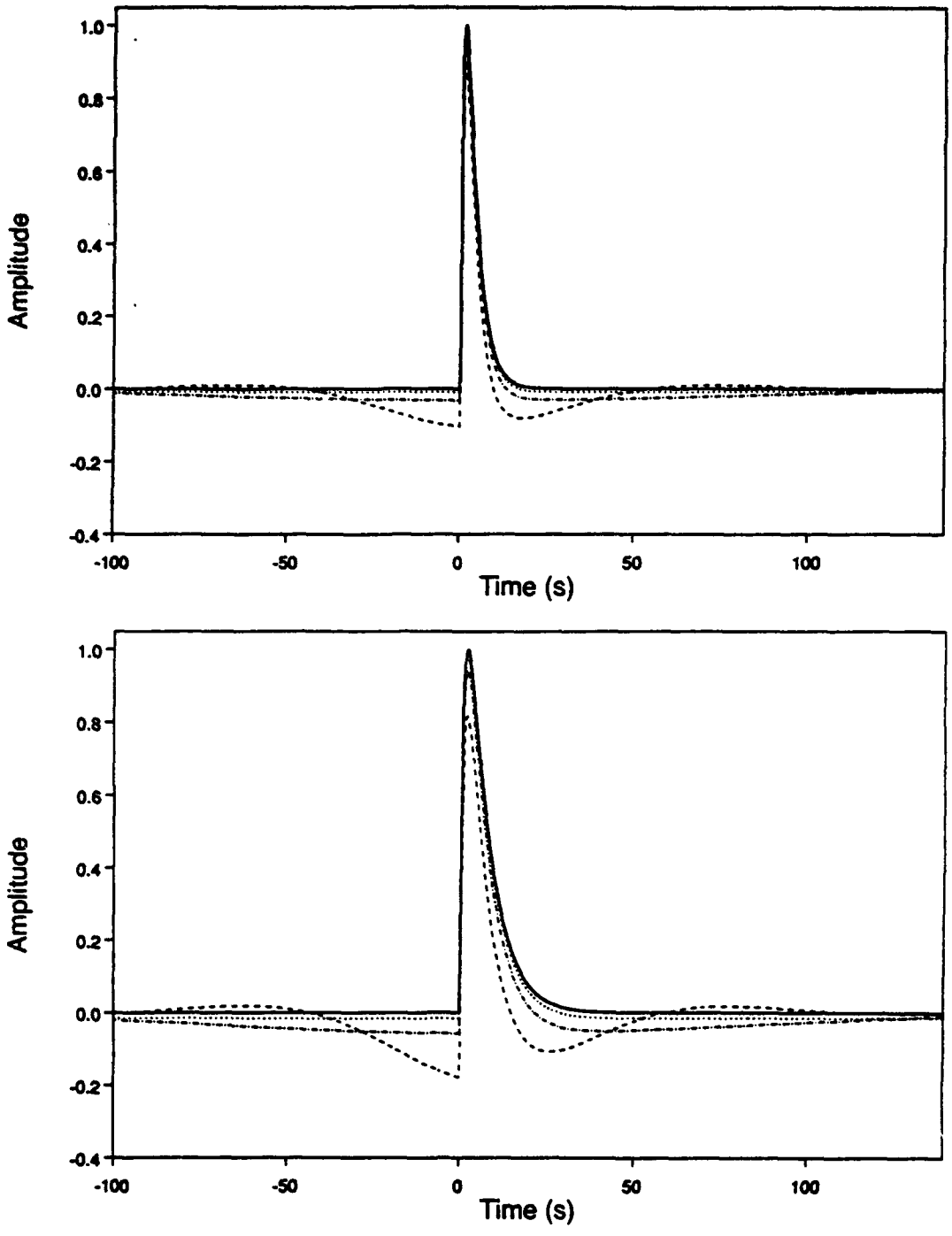


FIG. 2. Acausally filtered source functions $s(1, 3, t)$ (*top*) and $s(1, 6, t)$ (*bottom*). Original source functions (heavy solid line) and source functions after applying high-pass filters with corner frequencies of 0.001 Hz (dotted), 0.003 Hz (dot-dashed), and 0.01 Hz (dashed).

of the physical mechanisms invoked to explain these differences have been underestimated.

A different approach, used by Beck and Lay (1986), calculates the centroidal parameters using the square of the time series obtained through an acausal high-pass filter. Squaring the time series emphasizes the higher frequencies, so

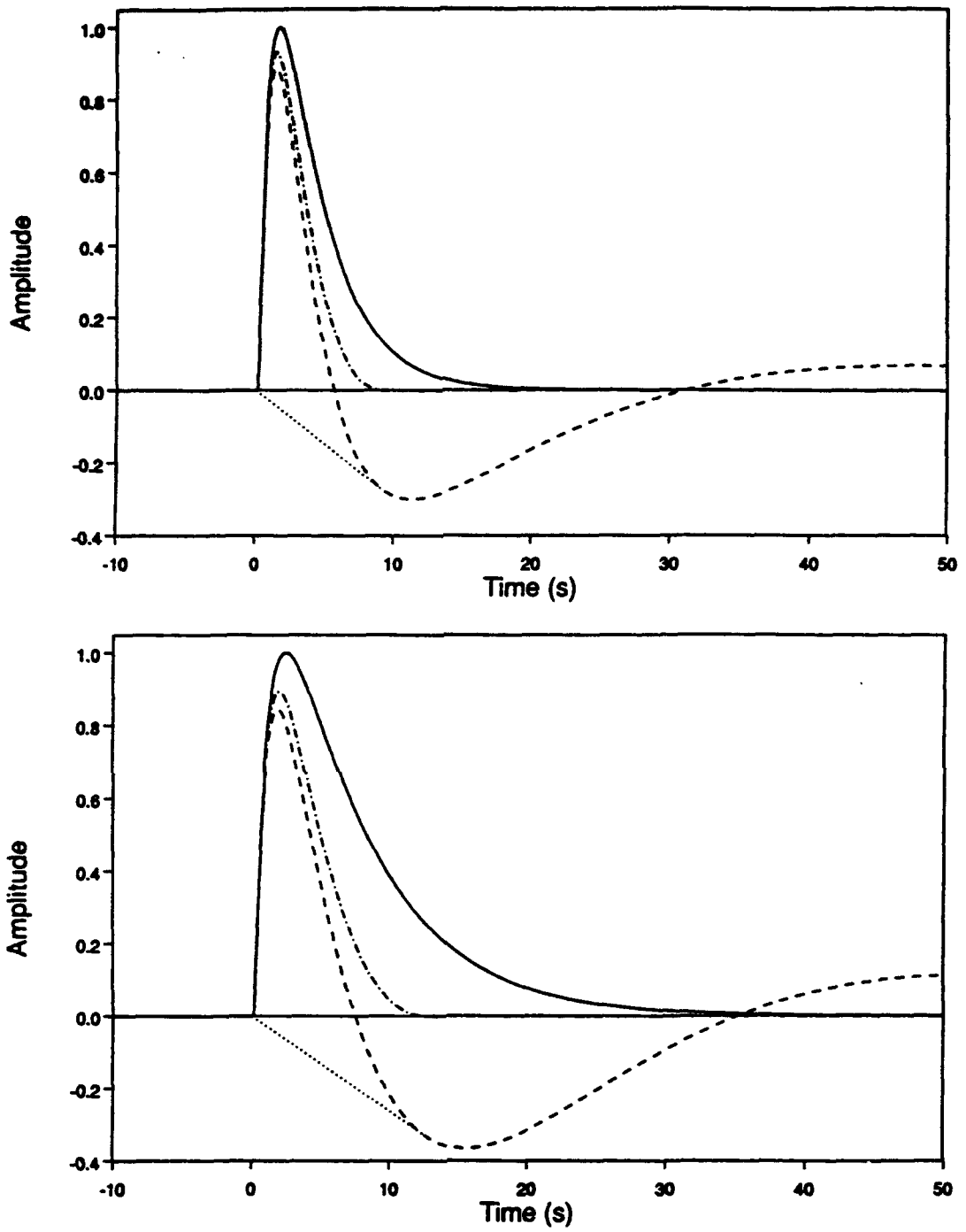


FIG. 3. "Detrending" procedure. The original unfiltered source functions $s(1, 3, t)$ (*top*) and $s(1, 6, t)$ (*bottom*) are shown as solid lines; after applying a causal high-pass filter with a corner frequency of 0.01 Hz, the time series are shown as dashed lines; the trends are shown as dotted lines; after subtracting the trends from the filtered time series, the results are shown as dot-dashed lines.

we expect that the biases will not be as large as those found using the previous procedure. However, a higher corner frequency of 0.025 Hz was used, which may be expected to increase the bias. The effect of applying this procedure to the waveforms in Figure 2 is shown in Figure 5; again, both the centroid and

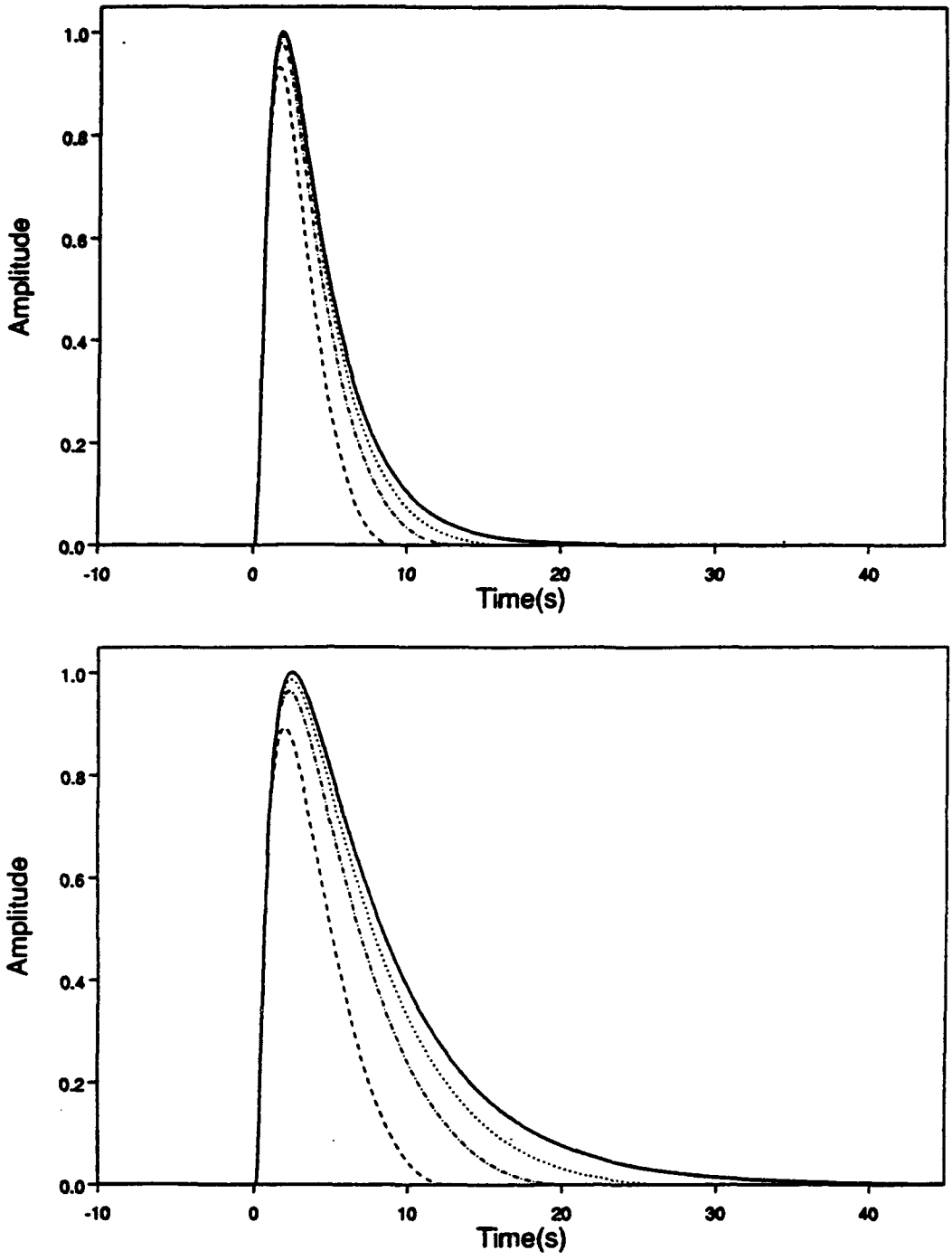


FIG. 4. "Detrended" source functions $s(1, 3, t)$ (*top*) and $s(1, 6, t)$ (*bottom*), with no filtering (heavy solid line) and after applying causal high-pass filters with corner frequencies of 0.001 Hz (dotted), 0.003 Hz (dot-dashed), and 0.01 Hz (dashed).

the variance of the pulses are affected, especially when using the higher corner frequencies. The centroidal parameters are listed in Table 2. The biases introduced, while not as large as those discussed in the previous paragraph, are still significant. In particular, for a corner frequency of 0.025 Hz, the true centroidal difference between the 3 and 6 sec pulses is 2.0 sec, but the measured difference

TABLE 1

CENTROIDAL PARAMETERS OF DETRENDED CAUSAL PULSES

f_c (Hz)	μ_0 Area	μ_1/μ_0 Centroid (sec)	μ_2/μ_0 Variance (sec^2)
τ_2 = 3 sec			
Unfiltered	5.13	4.26	10.42
0.001	4.72	3.79	6.52
0.003	4.24	3.39	4.60
0.010	3.26	2.72	2.44
τ_2 = 6 sec			
Unfiltered	9.26	7.62	39.39
0.001	8.01	6.24	19.51
0.003	6.72	5.22	11.85
0.010	4.47	3.73	4.93

will be 0.9 sec. The bias of 1.1 sec is on the order of the largest differences measured by Beck and Lay (1986). Again, this implies that differences reported in any studies using this procedure are too low.

Synthetic tests on unrealistic, symmetric source functions can lead to misleading results. For example, to evaluate the effect of their "detrending" procedure, Silver and Masuda (1985) performed tests of their deconvolution using synthetic data consisting of boxcar source-time functions of different lengths. They found that the high-pass filter introduced a bias in the variance of 3.0 sec^2, independent of source duration. This finding is an artifact of using a symmetric source function. As we have shown using asymmetric input pulses, the bias strongly depends on the duration of the pulse. In order to investigate the effects of his causal filters, Ekström (1987) inspected the impulse response of his system, i.e., he tested his deconvolution procedure using a delta-function source-time function. As a result of the symmetry of the source function, as well as the lack of power at low frequencies, the resulting filtered pulses also showed very little asymmetry.

The third-order Butterworth filters used in these examples are relatively gradual filters and in some studies sharper filters are used. Figure 6 shows the source function $s(1, 6, t)$, both unfiltered and after being high-pass filtered with a third-order and sixth-order acausal Butterworth filter, both with corner frequencies of 0.01 Hz. Using a sharper filter results in a narrower pulse with larger side-lobes, thus distorting the waveforms even more than those shown in the foregoing examples.

INSTRUMENT DECONVOLUTION USING REGULARIZATION METHODS

The results of the previous sections indicate that *a posteriori* filtering of the deconvolved displacement may be unwise. The *a posteriori* filtering, normally a high-pass, is an effort to mitigate the instability introduced when deconvolving the impulse response of a velocity sensor. The situation becomes even more complicated when significant amounts of ambient noise are present in the observed signal, since the noise at low frequencies is amplified by the deconvolution.

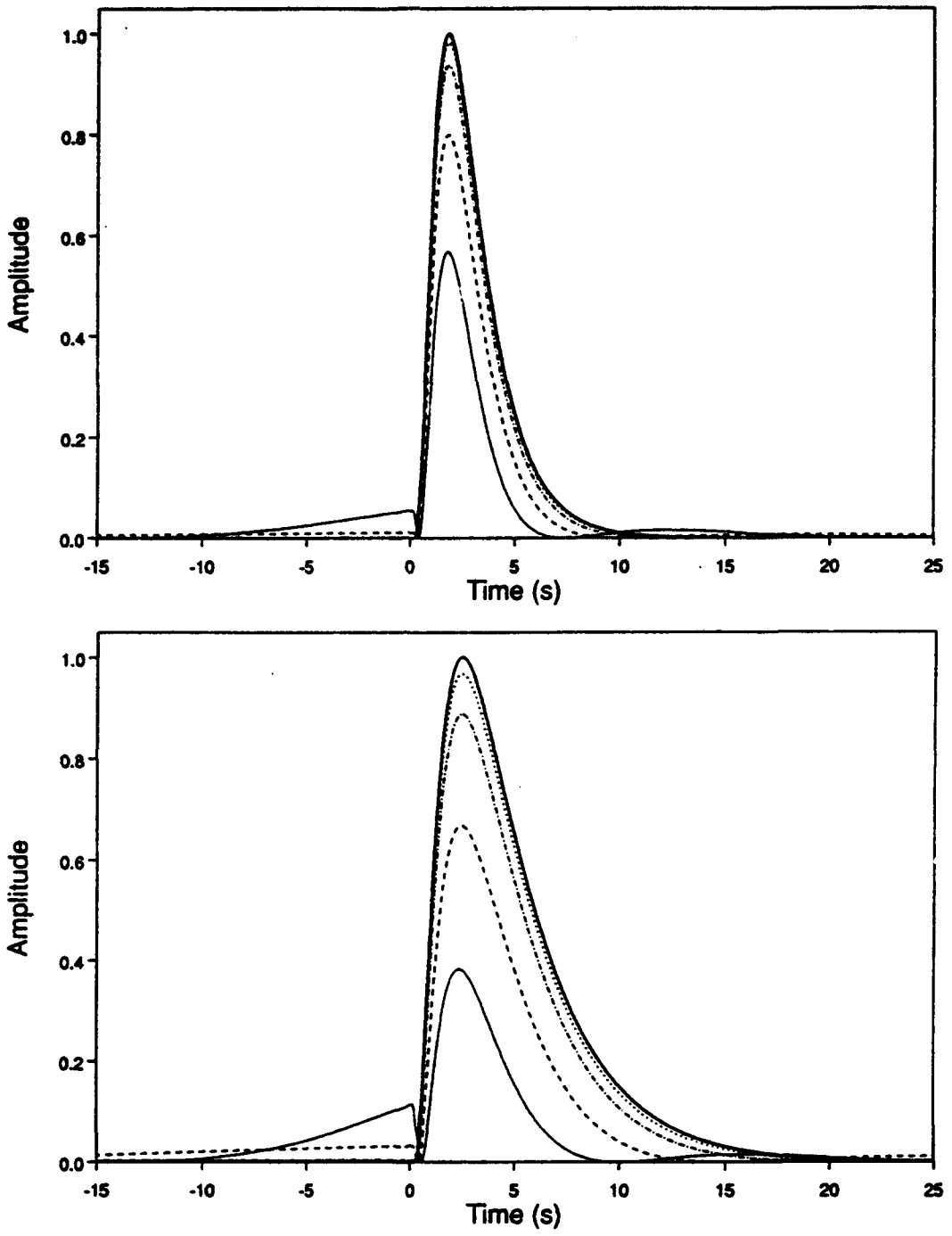


FIG. 5. Squared source functions $s(1, 3, t)$ (*top*) and $s(1, 6, t)$ (*bottom*), with no filtering (heavy solid line) and after applying acausal high-pass filters with corner frequencies of 0.001 Hz (dotted), 0.003 Hz (dot-dashed), 0.01 Hz (dashed), and 0.025 Hz (light solid line).

Deconvolution is usually cast as a solution to a Fredholm equation of the first kind (O'Dowd, 1990) with

$$d(t) = \int_{\tau_1}^{\tau_2} r(t - \tau)\, s(\tau)\, d\tau, \tag{2}$$

TABLE 2

CENTROIDAL PARAMETERS OF SQUARED ACAUSAL PULSES

f_c (Hz)	μ_0 Area	μ_1/μ_0 Centroid (sec)	μ_2/μ_0 Variance (sec^2)
τ_2 = 3 sec			
Unfiltered	3.27	2.94	3.16
0.001	3.18	2.91	3.06
0.003	2.97	2.86	2.97
0.010	2.36	2.72	2.57
0.025	1.45	2.47	1.62
τ_2 = 6 sec			
Unfiltered	5.76	4.92	11.31
0.001	5.46	4.80	10.00
0.003	4.79	4.56	8.04
0.010	3.14	4.03	4.93
0.025	1.39	3.34	2.41

where $r(t)$ is the convolutional kernel, $d(t)$ is the observed signal, and $s(t)$ is the deconvolved signal to be determined. If the signals $\mathbf{d}$ and $\mathbf{s}$ are sampled at times $\{t_i;\ i = 1,\cdots, N\}$ and the kernel r at times $\{t_i;\ i = 1,\cdots, M;\ M \leqq N\}$, then equation (2) can be written in operator form as

$$\mathbf{d} = \mathcal{R}\,\mathbf{s} \tag{3}$$

with $\mathbf{d}^t = [d(t_1),\cdots, d(t_N)]$, $\mathbf{s}^t = [s(t_1),\cdots, s(t_N)]$ and

$$\mathcal{R} = \begin{bmatrix} r_1 & r_2 & \cdots & r_M & 0 & \cdots & \cdots & 0 \\ 0 & r_1 & r_2 & \cdots & r_M & 0 & \cdots & 0 \\ 0 & 0 & r_1 & r_2 & \cdots & r_M & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & r_1 \end{bmatrix}, \tag{4}$$

where $\mathcal{R}$ is (ignoring end effects) square and Toeplitz. In the frequency domain, equation (3) is a simple filter multiplication and the deconvolution filter is just the reciprocal of the convolution filter. The time-domain formulation, however, lends itself more readily to treatment as a linear inverse problem, although frequency domain analogs do exist. Treating the deconvolution problem in equation (2) as a solution to the linear system in equation (3) is routine in exploration seismology, and many methods have been developed. In particular, equation (3) has been cast as a linear inverse problem by Oldenburg (1981). Our purpose here is to show how linear inverse methods can be applied in the case where $r(t)$ is the impulse response of a seismometer.

Equation (3) has a solution if the inverse of $\mathcal{R}$ exists and is bounded, which, as pointed out most recently by O'Dowd (1990), cannot be assumed in general. In this context, the deconvolution of instrument response is inherently ill posed. A velocity sensor must have zero response to DC input, making the frequency-domain deconvolutional filter singular at zero frequency. In addition to this

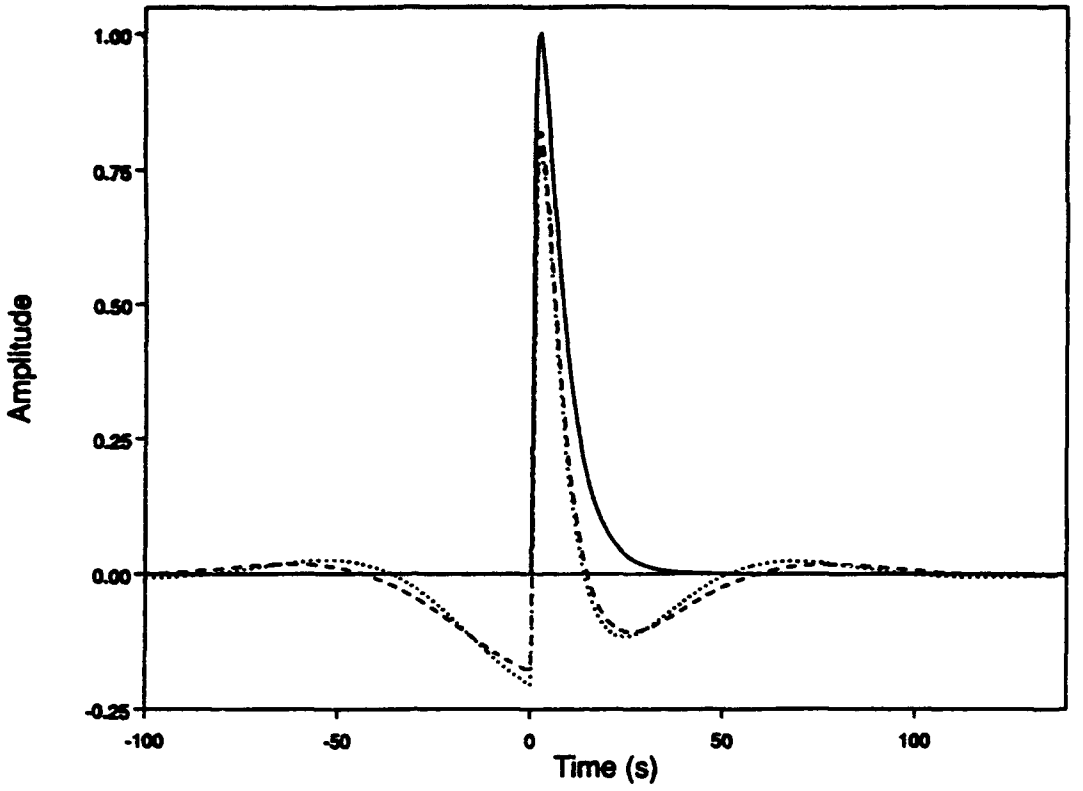


FIG. 6. Source function $s(1, 6, t)$ with no filtering (heavy solid line) and after applying third-order (dashed) and sixth-order (dotted) acausal Butterworth high-pass filters, both with corner frequencies of 0.01 Hz.

singularity, the falloff of response at frequencies below the instrument free period is rapid (exceeding 24 dB per octave for SRO instruments) making the deconvolutional filter poorly determined and, worse, an amplifier for ambient noise below the instrument corner. As we showed previously, this is the sort of noise amplification that can cause erroneous or biased measurements of pulse properties. This is usually mitigated in the frequency domain by *ad hoc* modification of the instrument response at about 40 dB below the corner or by applying a post-deconvolution high-pass filter (a "postwhitening"). These procedures are inadequate when the deconvolved noise is sufficiently red with respect to the fundamental period of the deconvolved signal, as is usually observed.

In the time domain, the impulse response of a velocity sensor must have zero mean. Thus the columns of $\mathcal{R}$ are linearly dependent and $\mathcal{R}$ is strictly singular. In the finite-dimensional discrete case, $\mathcal{R}$ is at best ill conditioned, depending on the length of the impulse response and the manner in which the endpoints of the convolution are handled. Solution of equation (3) by standard Toeplitz recursion (an N-squared process) is thus unstable since pivoting is not permitted (Press *et al.*, 1987), and a generalized inverse $\mathcal{R}^{\dagger}$ must be constructed (usually at least an N-cubed process). Since $\mathcal{R}$ is rank-deficient, $\mathcal{R}^T\mathcal{R}$ is also singular, and the "normal" solution

$$\mathbf{s} = \mathcal{R}^{\dagger}\mathbf{d} \tag{5}$$

with

$$\mathcal{R}^{\dagger} = \left(\mathcal{R}^T\mathcal{R}\right)^{-1}\mathcal{R}^T, \tag{6}$$

which solves the least-squares condition

$$\|\mathcal{R}\mathbf{s} - \mathbf{d}\|^2 = \text{minimum}, \tag{7}$$

is also unstable. Regularization methods (Jackson, 1972; Parker, 1972) thus are required. While generalized inverse methods are generally more costly than Toeplitz or frequency-domain methods, they offer the advantage that constraints on the solution, for the purpose of noise reduction or prewhitening, can be incorporated as regularizations in a natural way.

The purpose of regularizing the generalized inverse is to place a lower bound on the singular values of $\mathcal{R}$ so that $\mathcal{R}^{\dagger}$ is bounded. The method of singular value truncation (Parker 1972; Press *et al.*, 1987) applied in the case of instrument deconvolution is tantamount to filtering the solution with a box-car (sharp-edged) low-pass, with its concomitant edge effect. It is more useful, and far less *ad hoc*, to modify the distribution of small singular values by using estimates of the data variance and *a priori* constraints in the manner of Franklin (1970) or Tarantola and Valette (1982) (see also Lerner-Lam and Jordan, 1987). For example, this can be done by finding the solution to

$$\|\mathcal{R}\mathbf{s} - \mathbf{d}\|^2 + \alpha^2\|\mathbf{s}\|^2 = \text{minimum}, \tag{8}$$

namely

$$\mathcal{R}^{\dagger} = \left(\mathcal{R}^T\mathcal{R} + \alpha^2\mathbf{I}\right)^{-1}\mathcal{R}^T, \tag{9}$$

where α^2 can be interpreted as the ratio of the data variance to the model variance (Jordan and Franklin, 1971). This is the approach taken, for example, by O'Dowd (1990) for the regularization of autoregressive prediction. This regularization of the generalized inverse still has the effect of filtering the solution with a low pass, but the corner of the low-pass is smoother and controlled by the variance ratio and not an arbitrary eigenvalue cutoff.

The behavior of the deconvolution filter can be more explicitly regularized by requiring that the solution fit a linear constraint to within a specified variance. In this case, we designate a linear operator $\mathcal{B}$ so that $\mathcal{B}\mathbf{s} = \mathbf{b}$. The constraint vector $\mathbf{b}$ is assumed to have a distribution with expected value $\mathbf{b}_0$ and covariance $\mathbf{C}_{bb} = \sigma_b^2\mathbf{I}$. It can be shown (see Appendix) that the solution to

$$\|\mathcal{R}\mathbf{s} - \mathbf{d}\|^2 + \alpha^2\|\mathbf{s}\|^2 + \beta^2\|\mathcal{B}\mathbf{s} - \mathbf{b}_0\|^2 = \text{minimum} \tag{10}$$

is

$$\mathcal{R}^{\dagger} = \left(\mathcal{R}^T\mathcal{R} + \alpha^2\mathbf{I} + \beta^2\mathcal{B}^T\mathcal{B}\right)^{-1}\mathcal{R}^T, \tag{11}$$

where β^2 is the ratio of the data variance to the constraint variance. Instead of forming the generalized inverse in equation (11) explicitly, we prepare a new linear system

$$\mathbf{d}' = \mathcal{R}'\mathbf{s} \tag{12}$$

with

$$\mathbf{d}' = \begin{bmatrix} \mathbf{d} \\ 0 \\ \mathbf{b}_0 \end{bmatrix}$$

$$\mathscr{R}' = \begin{bmatrix} \mathscr{R} \\ \alpha^2 \mathbf{I} \\ \beta^2 \mathscr{B} \end{bmatrix} \tag{13}$$

and solve equation (12) by singular value decomposition (Lawson and Hanson, 1974).

The problem thus reduces to the appropriate choice of operator $\mathscr{B}$, constraint $\mathbf{b}_0$, and the variance ratios α and β. For example, smoothing and roughening constraints can be applied by forming the operators (Lerner-Lam, 1982)

$$\mathscr{B}_{\text{smooth}} = \begin{bmatrix} \ddots & \ddots & \ddots & & & 0 \\ & \ddots & \ddots & \ddots & & \\ & 1 & -2 & 1 & & \\ & & 1 & -2 & 1 & \\ 0 & & & \ddots & \ddots & \ddots \end{bmatrix}$$

$$\mathscr{B}_{\text{rough}} = \begin{bmatrix} 1 & & 0 \\ \vdots & \ddots & \\ 1 & \cdots & 1 \end{bmatrix} \tag{14}$$

$$\mathbf{b}_0 = 0,$$

where the rows of $\mathscr{B}_{\text{smooth}}$ and $\mathscr{B}_{\text{rough}}$ are just the convolution kernels for the discrete second derivative or cumulant, respectively. In practice, the values of α and β are chosen to be several orders of magnitude larger than the ratio of the smallest to the largest singular value of $\mathscr{R}$, $\lambda_{\text{min}}/\lambda_{\text{max}}$, which itself is limited by machine precision.

The smoothing and roughening constraints operate like low- and high-pass filters on the solution and are sufficient to stabilize the instrument deconvolution problem. The choice of α and β can be made data-adaptive by generalizing these ratios to include the auto-correlation matrix of the data, as suggested by O'Dowd (1990). The formalism for this is a generalization of the "total inversion" formalism of Tarantola and Valette (1982) and is given in the Appendix. Here, we assume the data, model, and constraint vectors to be uncorrelated and to have constant diagonal auto-correlation matrices.

Tests with Synthetic Noisy Data

We tested the algorithm with source function $s(1, 6, t)$ contaminated with 10% red noise. This is a realistic approximation of the noise observed for moderate-sized earthquakes ($m_b \sim 5.8$). Red noise is computed by integrating white noise with a standard deviation of 10% of the raw amplitude. This is then added to the raw pulse. Figure 7 shows the pure and contaminated source functions together with their pass through a typical long-period GDSN instrument response. The effect of the noise on the "baseline" of the raw pulse is apparent in the underlying long-period trend, but, as expected, most evidence of noise has been removed by convolution with the instrument response. The

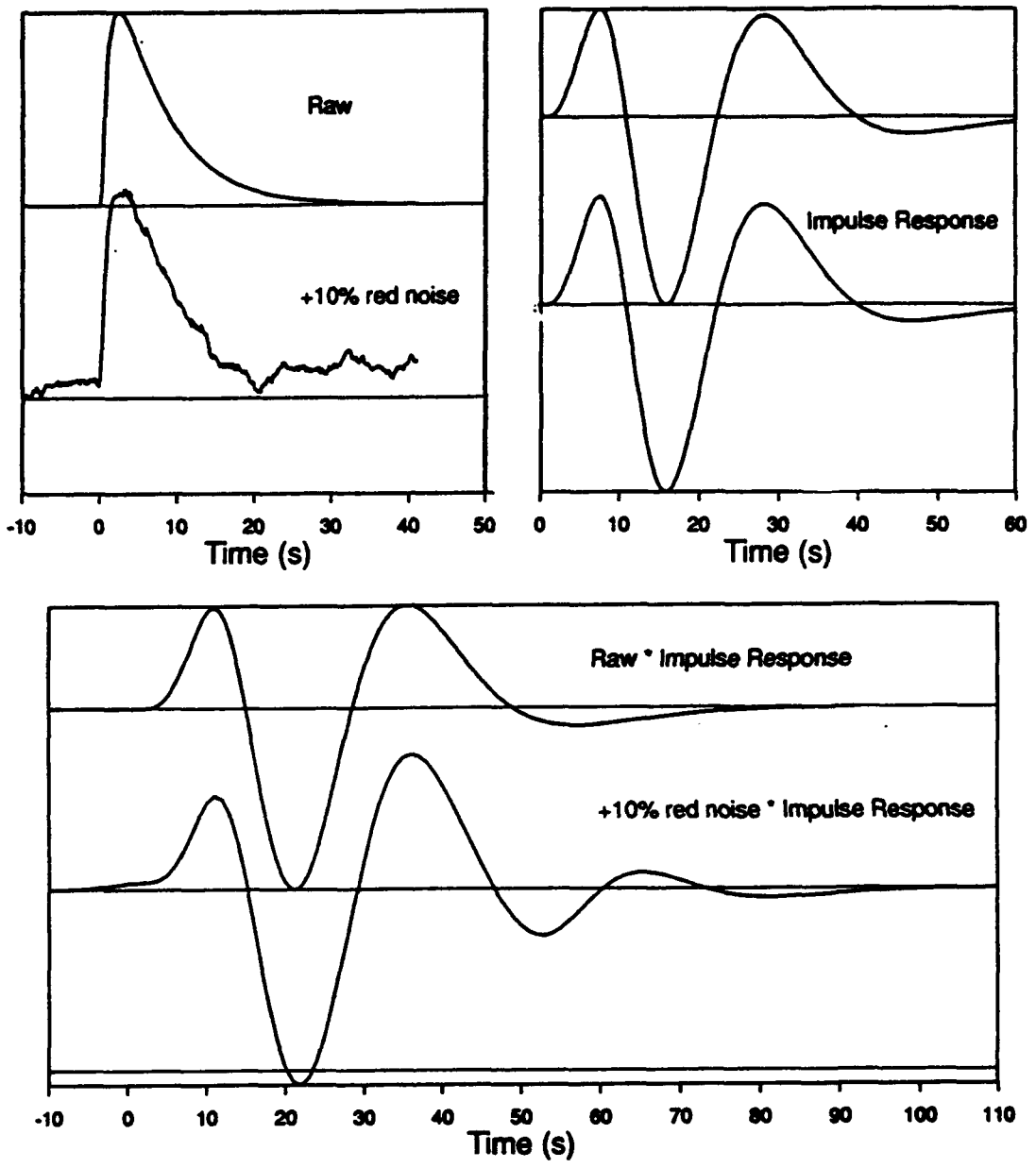


FIG. 7. Effect of convolution with instrument response on synthetic source function, with and without 10% red noise. (*Upper left*) Raw and contaminated source function $s(1, 6, t)$. The distortion of the baseline results from the large signals at periods large with respect to the length of the signal. (*Upper right*) Impulse response computed for a standard GDSN digital long-period instrument. (*Bottom*) Convolved raw and contaminated source functions.

addition of noise also causes a false extension, both at the beginning and at end of the pulse. We performed a "standard" deconvolution by constructing a deconvolution filter equal to the inverse of the instrument response in the frequency domain and treating the DC singularity by setting the deconvolution filter equal to zero for frequencies below the point at which the instrument response is 100 dB below its peak. The deconvolved signal was then high-pass filtered with a third-order causal Butterworth filter. These results are shown in Figure 8 for varying corner frequencies. Figure 8 shows that it is quite possible to mistake the long-period response of the high-pass filter for underlying

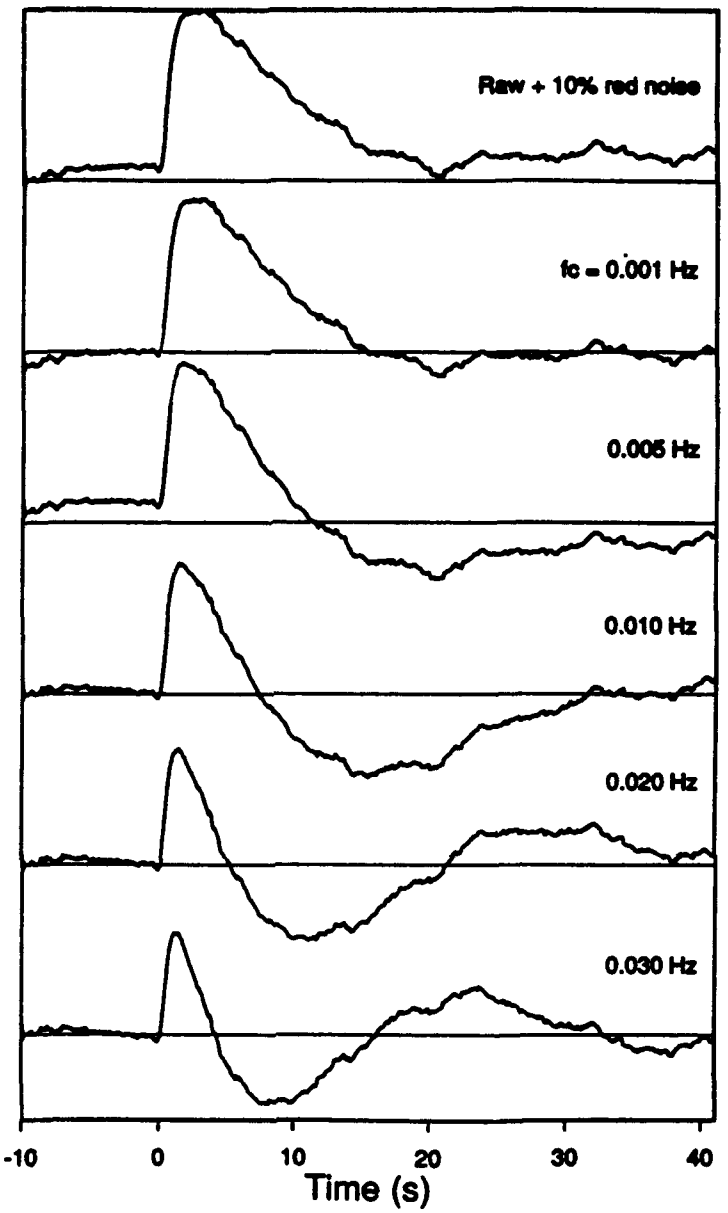


FIG. 8. Effect of high-pass filter applied after "standard" spectral instrument deconvolution. The deconvolution filter is constructed from the inverse of the instrument response in frequency. The singularity at DC is removed by setting the deconvolution filter to zero below the frequency where the instrument response has fallen to 100 dB below its peak. In no case is the deconvolved pulse a good match to the original.

long-period noise, resulting in a misleading baseline. The situation becomes more acute as the corner of the high-pass filter is moved to higher frequencies.

We next applied the optimum deconvolution filter produced by constrained singular-value decomposition of the Toeplitz response matrix. Figure 9 shows the results for the minimum-norm constraint (equation 8) with the ratio of the data variance to the model variance, α, set to 10^{-6}, 10^{-4}, and 10^{-2}. The deconvolution is stable and accurate for values of α below 10^{-2}, and there is no distortion of the baseline. When the data variance rises above 1% of the model variance (that is, 10% of the model standard deviation), some information at

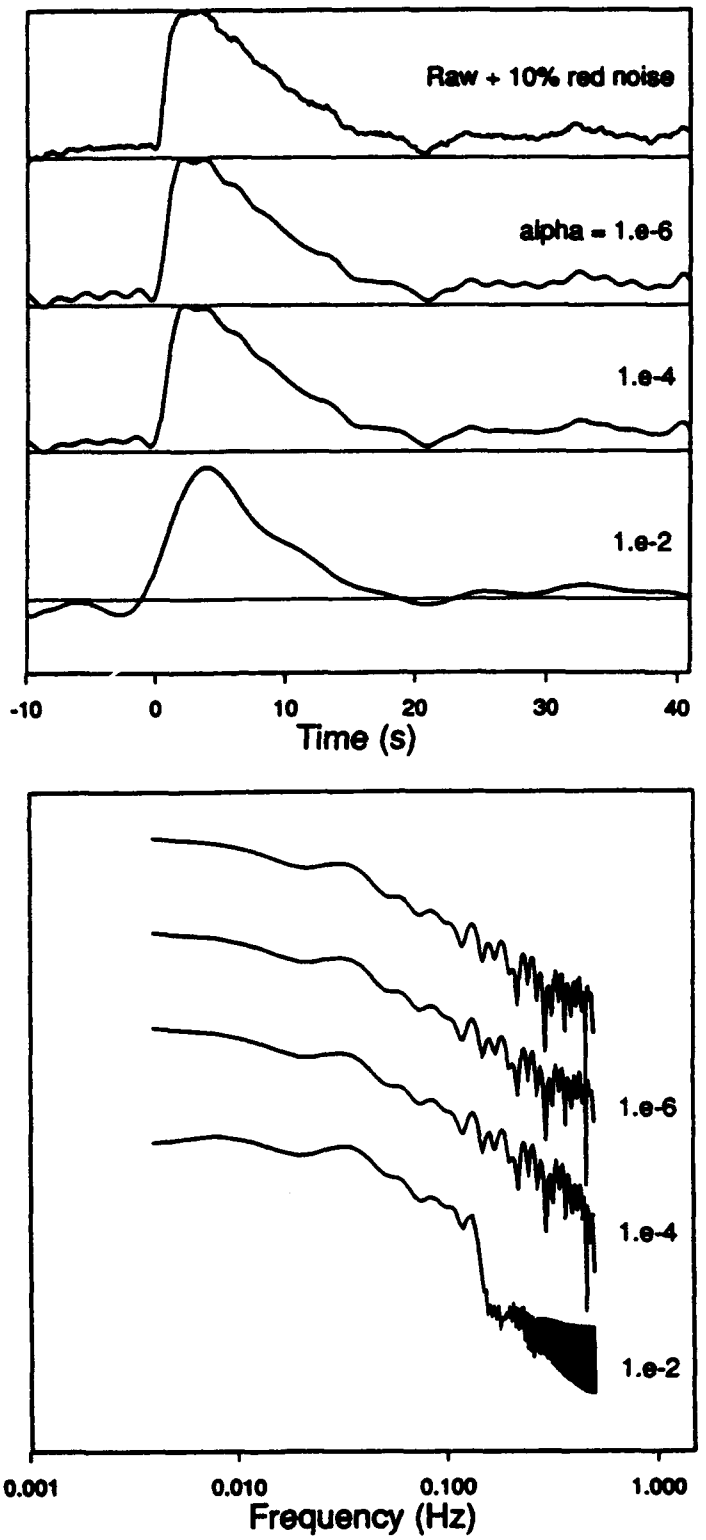


FIG. 9. Minimum-norm deconvolution shown in time (*top*) and frequency (*bottom*) domains, with the original signal shown in the top traces. Successive traces show the effect of increasing α, the ratio of the data variance to the model variance. The horizontal solid lines are the zero level for each trace. The effect of α is not noticeable until α attains a value consistent with the relative noise variance, above which information at the lowest and highest frequencies is lost.

long and short periods is lost from the signal, modifying the baseline and smoothing the waveform. This variance level is consistent with the level of the red noise added to the synthetic source function, suggesting that the method can be made data-adaptive by realistic assessment of the pre-arrival signal-to-noise ratio. Smoothness can also be enforced by minimizing the integral of the second derivative of the model (e.g., Lerner-Lam, 1982). Figure 10 shows the effect of incrementing the ratio of data variance to constraint variance, β, in equation (10), with α held fixed at 10^{-6}. Again, the deconvolution appears stable and accurate for values of β below the signal-to-noise ratio, but in this case there is less distortion of the long-period signal. The differences with respect to the minimum norm deconvolution are minimal however.

Finally, Figure 11 shows the effect of a roughening operator on the constrained deconvolution. The roughening operator is most like a high-pass filter in that it emphasizes higher frequencies. This effect can be seen in Figure 11 as a loss of DC information in the time series and spectra at relative variance levels above 1%, but the baseline is not otherwise distorted. This method, in effect, applies an "optimum" high-pass filter to the model, but the parameters of this filter are data-adaptive rather than *ad hoc*.

In order to assess the amount of bias introduced by this procedure when taking integral measures of the broadband displacement pulse, we have determined the area, centroid, and variance of the waveforms shown in Figure 9. These are listed in Table 3. The added noise has the expected effects: the area of the noisy source function is approximately 10% larger than that of the uncontaminated source function, and the centroid and variance are somewhat smaller. The latter is due to the fact that the red noise contributes characteristics to the waveform that make it appear more time-limited than the uncontaminated source function. The best that one can hope for, when taking integral measures of a deconvolved wavelet in the presence of red noise, is to replicate those obtained for the noisy source function. The centroidal parameters computed from the deconvolved noisy pulses differ from those of the original noisy pulse by only a few percent and are, in fact, much closer to those computed for the raw source function than those in Table 1.

Conclusions

The high-pass filtering part of the deconvolution procedure introduces a bias such that the measured centroidal parameters always have too low a value, and the bias is directly proportional to the duration of the pulse. The measured differences between centroidal parameters for pulses with differing durations are, therefore, also always biased to low values, with larger biases for pulse pairs having greater differences in their durations. This implies that the magnitudes of the mechanisms used to explain the observed differences, such as multipathing due to a high-velocity slab, slab diffraction, effects of lower-mantle heterogeneity, etc., have been underestimated.

There are several solutions to the problem of removing the bias in the centroidal parameters. One solution is to use data from broadband seismograph systems with better-calibrated transfer functions, such as GEOSCOPE and IRIS, and to use a high-pass filter with a lower corner frequency. As we have shown in the preceding section, however, there is still a significant amount of pulse distortion even when using very low-corner-frequency *a posteriori* filters. The distortion introduced by the filters will be irrelevant if the analysis is done

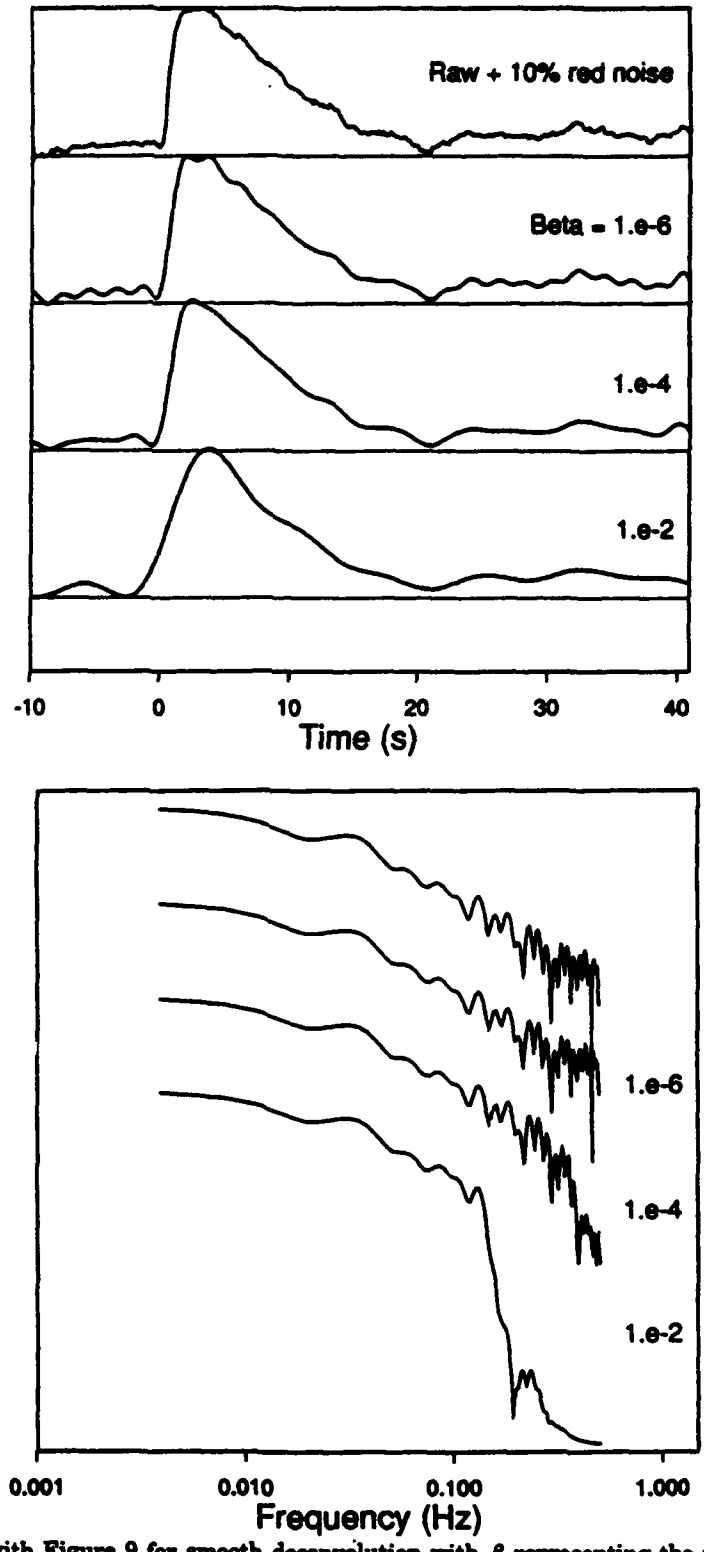


FIG. 10. As with Figure 9 for smooth deconvolution with β representing the ratio between the data variance and the constraint variance and α held fixed at 10^{-6}. Here the smoothing effect is limited to the highest frequencies present in the signal, as expected.

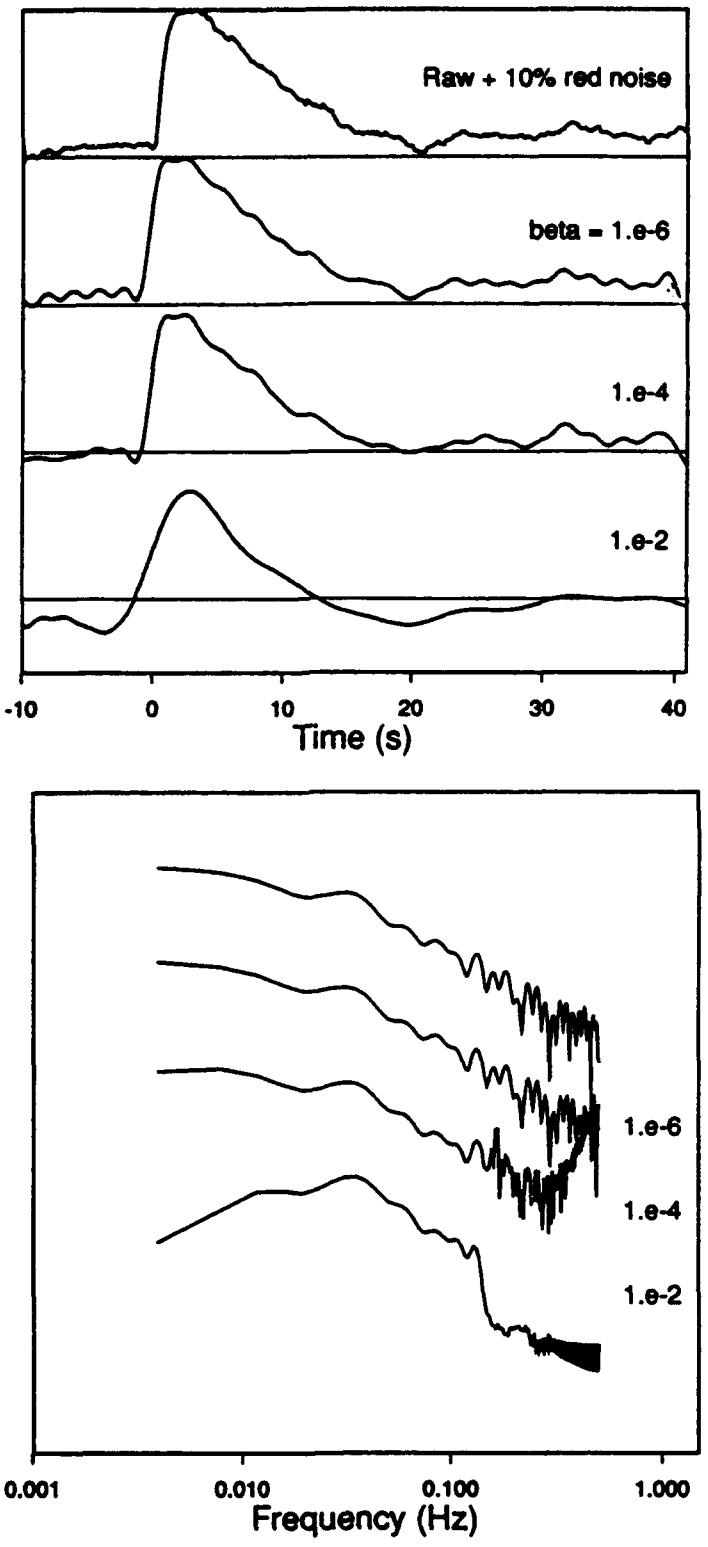


FIG. 11. As with Figures 9 and 10 for rough deconvolution, α held fixed at 10^{-6}. As β increases, energy at low frequencies is reduced and energy at high frequencies is enhanced. When β attains a value equal to the relative noise variance, almost all baseline information is lost. These results suggest that the procedure can be made data adaptive by proper assessment of the signal-to-noise ratio.

TABLE 3

CENTROIDAL PARAMETERS OF DECONVOLVED NOISY PULSES

α	μ_0 Area	μ_1/μ_0 Centroid (sec)	μ_2/μ_0 Variance (sec^2)
τ_2 = 6 sec			
Raw	9.26	7.62	39.39
+10% noise	10.70	6.98	21.59
10^{-6}	10.82	7.08	22.00
10^{-4}	10.82	7.07	22.00
10^{-2}	10.81	6.86	22.62

by comparing the observed seismograms to synthetic data in which the same filtering has been applied, as in the studies by Choy and Cormier (1986) and Vidale and Garcia-Gonzalez (1988). The parameters used in computing the synthetic seismograms, such as pulse duration, are known and can be directly compared.

In order to obtain the closest approximation to the true ground motion, however, the instrument response should be deconvolved using the regularization methods presented here. The proper construction of an instrument deconvolution filter is necessary for optimum estimation of pulse properties in source and propagation studies. The biases observed in this study can be mitigated by data-adaptive application of deconvolution filters constructed by regularized inverses. One is still left, however, with the need to accurately estimate a baseline in the presence of noisy data. Application of the optimum filters described here reproduces the long-period components of the signal with higher fidelity than do standard instrument deconvolution methods.

ACKNOWLEDGMENTS

We thank Tom Boyd for many amusing discussions and for his help in debugging some of the software used in this study and Eric Bergman, George Choy, William Menke, Paul Richards, and Chris Young for their reviews. A.L.L. thanks the USGS Branch of Global Seismology and Geomagnetism for use of its facilities and computers during the early parts of this study. Support at LDGO provided by AFGL F19628-89-K0007. LDGO contribution 4815.

REFERENCES

Beck, S. L. and T. Lay (1986). Test of the lower mantle slab penetration hypothesis using broadband *S* waves, *Geophys. Res. Lett.* **13**, 1007–1010.

Choy, G. L. and V. F. Cormier (1986). Direct measurement of the mantle attenuation operator from broadband *P* and *S* waveforms, *J. Geophys. Res.* **91**, 7326–7342.

Ekström, G. A. (1987). A broad band method of earthquake analysis, *Ph.D. Thesis*, Harvard University, Cambridge, Massachusetts.

Franklin, J. N. (1970). Well-posed stochastic extensions of ill-posed linear problems, *J. Math. Anal. Appl.* **31**, 682–716.

Futterman, W. I. (1962). Dispersive body waves, *J. Geophys. Res.* **67**, 5279–5291.

Harvey, D. and G. L. Choy (1982). Broad-band deconvolution of GDSN data, *Geophys. J.* **69**, 659–668.

Jackson, D. D. (1972). Interpretation of inaccurate, insufficient and inconsistent data, *Geophys. J.* **28**, 97–110.

Jordan, T. H. and J. N. Franklin (1971). Optimal solutions to a linear inverse problem in geophysics, *Proc. Natl. Acad. Sci.* **68**, 291–293.

Lawson, C. L. and D. J. Hanson (1974). *Solving Least Squares Problems*, Prentice Hall, Englewood Cliffs, New Jersey.

Lay, T. and C. J. Young (1989). Waveform complexity in teleseismic broadband *SH* displacements: slab diffractions or deep mantle reflections?, *Geophys. Res. Lett.* **16**, 605–608.

Lerner-Lam, A. L. (1982). Linearized estimation of higher-mode surface wave dispersion, *Ph. D. Thesis*, University of California, San Diego, California.

Lerner-Lam, A. L. and T. H. Jordan (1987). How thick are the continents?, *J. Geophys. Res.* **92**, 14,007–14,026.

O'Dowd, R. J. (1990). Ill-conditioning and pre-whitening in seismic deconvolution, *Geophys. J. Int.* **101**, 489–491.

Oldenburg, D. W. (1981). A comprehensive solution to the linear deconvolution problem, *Geophys J.* **65**, 331–357.

Parker, R. L. (1972). Inverse theory with grossly inadequate data, *Geophys. J.* **29**, 123–138.

Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vettering (1987). *Numerical Recipes in C*, Cambridge University Press, New York.

Seidl, D. and M. Hellweg (1988). Restoration of broad-band seismograms (Part II): signal moment determination, *J. Geophys.* **62**, 158–162.

Silver, P. G. and W. W. Chan (1986). Observations of body wave multipathing from broadband seismograms: evidence for lower mantle slab penetration beneath the Sea of Okhotsk, *J. Geophys. Res.* **91**, 13,787–13,802.

Silver, P. G. and T. Masuda (1985). A source extent analysis of the Imperial Valley earthquake of October 15, 1979, and the Victoria earthquake of June 9, 1980, *J. Geophys. Res.* **90**, 7639–7651.

Tarantola, A. and B. Valette (1982). Generalized nonlinear inverse problems solved using the least-squares criterion, *Rev. Geophys.* **20**, 219–232.

Vidale, J. E. and D. Garcia-Gonzalez (1988). Seismic observation of a high-velocity slab 1200–1600 km in depth, *Geophys. Res. Lett.* **15**, 369–372.

Appendix

The formalism presented by Tarantola and Valette (1982) allows general solution of linear (or linearized) inverse problems when the model, the data, and any side constraints on the model are covariant. We follow this formalism, expanding it to include constraint operators. Let $\mathbf{d}$ and $\mathbf{s}$ be the observed data and model signal, respectively; let $\mathscr{R}$ be the convolution (or other linear) operator defined in equation (2); and let $\mathscr{B}$ be a constraint operator such that $\mathscr{B}\mathbf{s} = \mathbf{b}$, where $\mathbf{b}$ is the value of the constraint. Define the vector $\mathbf{x}$ to be

$$\mathbf{x} = \begin{bmatrix} \mathbf{d} \\ \mathbf{s} \\ \mathbf{b} \end{bmatrix}. \tag{A1}$$

The vector $\mathbf{x}$ has the expected value $\mathbf{x}_0$

$$\mathbf{x}_0 = \begin{bmatrix} \mathbf{d}_0 \\ \mathbf{s}_0 \\ \mathbf{b}_0 \end{bmatrix} \tag{A2}$$

and covariance $\mathbf{C}_{xx}$

$$\mathbf{C}_{xx} = \begin{bmatrix} \mathbf{C}_{dd} & \mathbf{C}_{ds} & \mathbf{C}_{db} \\ \mathbf{C}_{sd} & \mathbf{C}_{ss} & \mathbf{C}_{sb} \\ \mathbf{C}_{bd} & \mathbf{C}_{bs} & \mathbf{C}_{bb} \end{bmatrix}, \tag{A3}$$

where $\mathbf{d}_0$, $\mathbf{s}_0$, and $\mathbf{b}_0$ are the expected values of $\mathbf{d}$, $\mathbf{s}$, and $\mathbf{b}$ and the submatrices in equation (A3) are the appropriate covariance matrices. $\mathbf{C}_{xx}$ is symmetric.

The Gaussian probability distribution function (PDF) for $\mathbf{x}$ is

$$\mathrm{PDF}(\mathbf{x}) = N_x \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{x}_0)^T \mathbf{C}_{xx}^{-1}(\mathbf{x}-\mathbf{x}_0)\right], \tag{A4}$$

where N_x is a suitable normalization and "T" denotes transpose. Define two operators:

$$\mathcal{F} = (\mathbf{I} \mid -\mathcal{R} \mid \mathbf{O}), \tag{A5a}$$

$$\mathcal{G} = (\mathbf{O} \mid -\mathcal{B} \mid \mathbf{I}), \tag{A5b}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{O}$ is a zero matrix of appropriate dimension. Then the equations $\mathcal{F}\mathbf{x} = 0$ and $\mathcal{G}\mathbf{x} = 0$ imply $\mathbf{d} = \mathcal{R}\mathbf{s}$ and $\mathbf{b} = \mathcal{B}\mathbf{s}$, respectively. The exponent of the PDF is minimized subject to the constraints $\mathcal{F}\mathbf{x} = 0$ and $\mathcal{G}\mathbf{x} = 0$ using the method of Lagrange multipliers, namely:

$$\min_{\mathbf{x}} = \left[(\mathbf{x}-\mathbf{x}_0)^T \mathbf{C}_{xx}^{-1}(\mathbf{x}-\mathbf{x}_0)\right] + \mathbf{x}^T\mathcal{F}^T\lambda + \mathbf{x}^T\mathcal{G}^T\gamma, \tag{A6}$$

where λ and γ are Lagrange multipliers. The solution, although tedious, is straightforward:

$$\mathbf{x} = \left[\mathbf{I} - \mathcal{P}_{\mathcal{G}}\mathbf{C}_{xx}\mathcal{F}^T\left(\mathbf{I} - \mathcal{F}^{\dagger}\mathcal{Q}_{\mathcal{G}}\mathbf{C}_{xx}\mathcal{F}^T\right)^{-1}\mathcal{F}^{\dagger}\right]\mathcal{P}_{\mathcal{G}}\mathbf{x}_0, \tag{A7a}$$

$$\mathcal{F}^{\dagger} = \left(\mathcal{F}\mathbf{C}_{xx}\mathcal{F}^T\right)^{-1}\mathcal{F}; \quad \mathcal{G}^{\dagger} = \left(\mathcal{G}\mathbf{C}_{xx}\mathcal{G}^T\right)^{-1}\mathcal{G}, \tag{A7b}$$

$$\mathcal{Q}_{\mathcal{G}} = \mathbf{C}_{xx}\mathcal{G}^T\mathcal{G}^{\dagger}, \tag{A7c}$$

$$\mathcal{P}_{\mathcal{G}} = \mathbf{I} - \mathcal{Q}_{\mathcal{G}}, \tag{A7d}$$

where "†" denotes a weighted generalized inverse and $\mathcal{P}_{\mathcal{G}}$ and $\mathcal{Q}_{\mathcal{G}}$ are orthogonal projection operators projecting onto the range and null space of $\mathcal{G}$. The additional complexity of equation (A7) over the results of Tarantola and Valette (1982) arises from the existence of the constraint operator $\mathcal{B}$. It should be noted that the quantity within the parentheses in equation (A7a) is itself a projection operator with an inverse that is usually unstable; generalized inverse methods should thus be used to compute it.

These equations are vastly simplified if we ignore the covariances among $\mathbf{d}$, $\mathbf{s}$, and $\mathbf{b}$ and assume that the submatrices $\mathbf{C}_{dd}$, $\mathbf{C}_{ss}$, and $\mathbf{C}_{bb}$ are diagonal:

$$\mathbf{C}_{dd} = \sigma_d^2\mathbf{I},$$

$$\mathbf{C}_{ss} = \sigma_s^2\mathbf{I},$$

$$\mathbf{C}_{bb} = \sigma_b^2\mathbf{I}. \tag{A8}$$

Equation (11) in the text is obtained from equation (A7) above in the limit as σ_s^2 and σ_b^2 become large with respect to σ_d^2 (model is weakly constrained).

U.S. GEOLOGICAL SURVEY
MS 967, BOX 25046, DFC
DENVER, COLORADO 80225
(S.A.S.)

LAMONT-DOHERTY GEOLOGICAL OBSERVATORY
PALISADES, NEW YORK 10964
(A.L.L.)

Manuscript received 10 January 1991

Prof. Thomas Ahrens
Seismological Lab, 252-21
Division of Geological & Planetary Sciences
California Institute of Technology
Pasadena, CA 91125

Prof. Keiiti Aki
Center for Earth Sciences
University of Southern California
University Park
Los Angeles, CA 90089-0741

Prof. Shelton Alexander
Geosciences Department
403 Deike Building
The Pennsylvania State University
University Park, PA 16802

Prof. Charles B. Archambeau
CIRES
University of Colorado
Boulder, CO 80309

Dr. Thomas C. Bache, Jr.
Science Applications Int'l Corp.
10260 Campus Point Drive
San Diego, CA 92121 (2 copies)

Prof. Muawia Barazangi
Institute for the Study of the Continent
Cornell University
Ithaca, NY 14853

Dr. Jeff Barker
Department of Geological Sciences
State University of New York
at Binghamton
Vestal, NY 13901

Dr. Douglas R. Baumgardt
ENSCO, Inc
5400 Port Royal Road
Springfield, VA 22151-2388

Dr. Susan Beck
Department of Geosciences
Building #77
University of Arizona
Tuscon, AZ 85721

Dr. T.J. Bennett
S-CUBED
A Division of Maxwell Laboratories
11800 Sunrise Valley Drive, Suite 1212
Reston, VA 22091

Dr. Robert Blandford
AFTAC/TT, Center for Seismic Studies
1300 North 17th Street
Suite 1450
Arlington, VA 22209-2308

Dr. Stephen Bratt
ARPA/NMRO
3701 North Fairfax Drive
Arlington, VA 22203-1714

Dr. Lawrence Burdick
IGPP, A-025
Scripps Institute of Oceanography
University of California, San Diego
La Jolla, CA 92093

Dr. Robert Burridge
Schlumberger-Doll Research Center
Old Quarry Road
Ridgefield, CT 06877

Dr. Jerry Carter
Center for Seismic Studies
1300 North 17th Street
Suite 1450
Arlington, VA 22209-2308

Dr. Eric Chael
Division ~~9241~~ MS-0655
Sandia Laboratory
Albuquerque, NM 87185-0655

Dr. Martin Chapman
Department of Geological Sciences
Virginia Polytechnical Institute
21044 Derring Hall
Blacksburg, VA 24061

Mr Robert Cockerham
Arms Control & Disarmament Agency
320 21st Street North West
Room 5741
Washington, DC 20451,

Prof. Vernon F. Cormier
Department of Geology & Geophysics
U-45, Room 207
University of Connecticut
Storrs, CT 06268

Prof. Steven Day
Department of Geological Sciences
San Diego State University
San Diego, CA 92182

US Dept of Energy
Recipient, IS-20, GA-033
Office of Rsch & Development
1000 Independence Ave
Washington, DC 20585

Dr. Zoltan Der
ENSCO, Inc.
5400 Port Royal Road
Springfield, VA 22151-2388

Prof. Adam Dziewonski
Hoffman Laboratory, Harvard University
Dept. of Earth Atmos. & Planetary Sciences
20 Oxford Street
Cambridge, MA 02138

Prof. John Ebel
Department of Geology & Geophysics
Boston College
Chestnut Hill, MA 02167

Eric Fielding
SNEE Hall
INSTOC
Cornell University
Ithaca, NY 14853

Dr. Petr Firbas
Institute of Physics of the Earth
Masaryk University Brno
Jecna 29a
612 46 Brno, Czech Republic

Dr. Mark D. Fisk
Mission Research Corporation
735 State Street
P.O. Drawer 719
Santa Barbara, CA 93102

Prof Stanley Flatte
Applied Sciences Building
University of California, Santa Cruz
Santa Cruz, CA 95064

Prof. Donald Forsyth
Department of Geological Sciences
Brown University
Providence, RI 02912

Dr. Art Frankel
U.S. Geological Survey
922 National Center
Reston, VA 22092

Dr. Cliff Frolich
Institute of Geophysics
8701 North Mopac
Austin, TX 78759

Dr. Holly Given
IGPP, A-025
Scripps Institute of Oceanography
University of California, San Diego
La Jolla, CA 92093

Dr. Jeffrey W. Given
SAIC
10260 Campus Point Drive
San Diego, CA 92121

Dr. Dale Glover
Defense Intelligence Agency
ATTN: ODT-1B
Washington, DC 20301

Dan N. Hagedon
Pacific Northwest Laboratories
Battelle Boulevard
Richland, WA 99352

Dr. James Hannon
Lawrence Livermore National Laboratory
P.O. Box 808
L-205
Livermore, CA 94550

Prof. David G. Harkrider
Seismological Laboratory
Division of Geological & Planetary Sciences
California Institute of Technology
Pasadena, CA 91125

Prof. Danny Harvey
CIRES
University of Colorado
Boulder, CO 80309

Prof. Donald V. Helmberger
Seismological Laboratory
Division of Geological & Planetary Sciences
California Institute of Technology
Pasadena, CA 91125

Prof. Eugene Herrin
Institute for the Study of Earth and Man
Geophysical Laboratory
Southern Methodist University
Dallas, TX 75275

Prof. Robert B. Herrmann
Department of Earth & Atmospheric Sciences
St. Louis University
St. Louis, MO 63156

Prof. Lane R. Johnson
Seismographic Station
University of California
Berkeley, CA 94720

Prof. Thomas H. Jordan
Department of Earth, Atmospheric &
Planetary Sciences
Massachusetts Institute of Technology
Cambridge, MA 02139

Prof. Alan Kafka
Department of Geology & Geophysics
Boston College
Chestnut Hill, MA 02167

Robert C. Kemerait
ENSCO, Inc.
445 Pineda Court
Melbourne, FL 32940

Dr. Karl Koch
Institute for the Study of Earth and Man
Geophysical Laboratory
Southern Methodist University
Dallas, Tx 75275

US Dept of Energy
Attn: Max Koontz, NN-20, GA-033
Office of Rsch & Development
1000 Independence Ave, SW
Washington, DC 20585

Dr. Richard LaCoss
MIT Lincoln Laboratory, M-200B
P.O. Box 73
Lexington, MA 02173-0073

Dr. Fred K. Lamb
University of Illinois at Urbana-Champaign
Department of Physics
1110 West Green Street
Urbana, IL 61801

Prof. Charles A. Langston
Geosciences Department
403 Deike Building
The Pennsylvania State University
University Park, PA 16802

Jim Lawson, Chief Geophysicist
Oklahoma Geological Survey
Oklahoma Geophysical Observatory
P.O. Box 8
Leonard, OK 74043-0008

Prof. Thorne Lay
Institute of Tectonics
Earth Science Board
University of California, Santa Cruz
Santa Cruz, CA 95064

Dr. William Leith
U.S. Geological Survey
Mail Stop 928
Reston, VA 22092

Mr. James F. Lewkowicz
Phillips Laboratory/GPEH
29 Randolph Road
Hanscom AFB, MA 01731-3010(2 copies)

Mr. Alfred Lieberman
ACDA/VI-OA State Department Building
Room 5726
320-21st Street, NW
Washington, DC 20451

Prof. L. Timothy Long
School of Geophysical Sciences
Georgia Institute of Technology
Atlanta, GA 30332

Dr. Randolph Martin, III
New England Research, Inc.
76 Olcott Drive
White River Junction, VT 05001

Dr. Robert Masse
Denver Federal Building
Box 25046, Mail Stop 967
Denver, CO 80225

Dr. Gary McCartor
Department of Physics
Southern Methodist University
Dallas, TX 75275

Prof. Thomas V. McEvilly
Seismographic Station
University of California
Berkeley, CA 94720

Dr. Art McGarr
U.S. Geological Survey
Mail Stop 977
U.S. Geological Survey
Menlo Park, CA 94025

Dr. Keith L. McLaughlin
S-CUBED
A Division of Maxwell Laboratory
P.O. Box 1620
La Jolla, CA 92038-1620

Stephen Miller & Dr. Alexander Florence
SRI International
333 Ravenswood Avenue
Box AF 116
Menlo Park, CA 94025-3493

Prof. Bernard Minster
IGPP, A-025
Scripps Institute of Oceanography
University of California, San Diego
La Jolla, CA 92093

Prof. Brian J. Mitchell
Department of Earth & Atmospheric Sciences
St. Louis University
St. Louis, MO 63156

Mr. Jack Murphy
S-CUBED
A Division of Maxwell Laboratory
11800 Sunrise Valley Drive, Suite 1212
Reston, VA 22091 (2 Copies)

Dr. Keith K. Nakanishi
Lawrence Livermore National Laboratory
L-025
P.O. Box 808
Livermore, CA 94550

Prof. John A. Orcutt
IGPP, A-025
Scripps Institute of Oceanography
University of California, San Diego
La Jolla, CA 92093

Prof. Jeffrey Park
Kline Geology Laboratory
P.O. Box 6666
New Haven, CT 06511-8130

Dr. Howard Patton
Lawrence Livermore National Laboratory
L-025
P.O. Box 808
Livermore, CA 94550

Dr. Frank Pilotte
HQ AFTAC/TT
1030 South Highway A1A
Patrick AFB, FL 32925-3002

Dr. Jay J. Pulli
Radix Systems, Inc.
201 Perry Parkway
Gaithersburg, MD 20877

Dr. Robert Reinke
ATTN: FCTVTD
Field Command
Defense Nuclear Agency
Kirtland AFB, NM 87115

Prof. Paul G. Richards
Lamont-Doherty Geological Observatory
of Columbia University
Palisades, NY 10964

Mr. Wilmer Rivers
Teledyne Geotech
314 Montgomery Street
Alexandria, VA 22314

Dr. Alan S. Ryall, Jr.
ARPA/NMRO
3701 North Fairfax Drive
Arlington, VA 22203-1714

Dr. Richard Sailor
TASC, Inc.
55 Walkers Brook Drive
Reading, MA 01867

Prof. Charles G. Sammis
Center for Earth Sciences
University of Southern California
University Park
Los Angeles, CA 90089-0741

Prof. Christopher H. Scholz
Lamont-Doherty Geological Observatory
of Columbia University
Palisades, NY 10964

Dr. Susan Schwartz
Institute of Tectonics
1156 High Street
Santa Cruz, CA 95064

Secretary of the Air Force
(SAFRD)
Washington, DC 20330

Office of the Secretary of Defense
DDR&E
Washington, DC 20330

Thomas J. Sereno, Jr.
Science Application Int'l Corp.
10260 Campus Point Drive
San Diego, CA 92121

Dr. Michael Shore
Defense Nuclear Agency/SPSS
6801 Telegraph Road
Alexandria, VA 22310

Dr. Robert Shumway
University of California Davis
Division of Statistics
Davis, CA 95616

Dr. Matthew Sibol
Virginia Tech
Seismological Observatory
4044 Derring Hall
Blacksburg, VA 24061-0420

Prof. David G. Simpson
IRIS, Inc.
1616 North Fort Myer Drive
Suite 1050
Arlington, VA 22209

Donald L. Springer
Lawrence Livermore National Laboratory
L-025
P.O. Box 808
Livermore, CA 94550

Dr. Jeffrey Stevens
S-CUBED
A Division of Maxwell Laboratory
P.O. Box 1620
La Jolla, CA 92038-1620

Lt. Col. Jim Stobie
ATTN: AFOSR/NL
110 Duncan Avenue
Bolling AFB
Washington, DC 20332-0001

Brian Stump
Los Alamos National Laboratory
EES-3, Mail Stop C335
Los Alamos, NM 87545

Prof. Jeremiah Sullivan
University of Illinois at Urbana-Champaign
Department of Physics
1110 West Green Street
Urbana, IL 61801

Prof. L. Sykes
Lamont-Doherty Geological Observatory
of Columbia University
Palisades, NY 10964

Dr. David Taylor
ENSCO, Inc.
445 Pineda Court
Melbourne, FL 32940

Dr. Steven R. Taylor
Los Alamos National Laboratory
P.O. Box 1663
Mail Stop C335
Los Alamos, NM 87545

Prof. Clifford Thurber
University of Wisconsin-Madison
Department of Geology & Geophysics
1215 West Dayton Street
Madison, WS 53706

Prof. M. Nafi Toksoz
Earth Resources Lab
Massachusetts Institute of Technology
42 Carleton Street
Cambridge, MA 02142

Dr. Larry Turnbull
CIA-OSWR/NED
Washington, DC 20505

Dr. Gregory van der Vink
IRIS, Inc.
1616 North Fort Myer Drive
Suite 1050
Arlington, VA 22209

Dr. Karl Veith
EG&G
5211 Auth Road
Suite 240
Suitland, MD 20746

Prof. Terry C. Wallace
Department of Geosciences
Building #77
University of Arizona
Tuscon, AZ 85721

Dr. Thomas Weaver
Los Alamos National Laboratory
P.O. Box 1663
Mail Stop C335
Los Alamos, NM 87545

Dr. William Wortman
Mission Research Corporation
8560 Cinderbed Road
Suite 700
Newington, VA 22122

Prof. Francis T. Wu
Department of Geological Sciences
State University of New York
at Binghamton
Vestal, NY 13901

ARPA, OASB/Library
3701 North Fairfax Drive
Arlington, VA 22203-1714

HQ DNA
ATTN: Technical Library
Washington, DC 20305

Defense Intelligence Agency
Directorate for Scientific & Technical Intelligence
ATTN: DTIB
Washington, DC 20340-6158

Defense Technical Information Center
Cameron Station
Alexandria, VA 22314 (2 Copies)

TACTEC
Battelle Memorial Institute
505 King Avenue
Columbus, OH 43201 (Final Report)

Phillips Laboratory
ATTN: XPG
29 Randolph Road
Hanscom AFB, MA 01731-3010

Phillips Laboratory
ATTN: GPE
29 Randolph Road
Hanscom AFB, MA 01731-3010

Phillips Laboratory
ATTN: TSML
5 Wright Street
Hanscom AFB, MA 01731-3004

Phillips Laboratory
ATTN: PL/SUL
3550 Aberdeen Ave SE
Kirtland, NM 87117-5776 (2 copies)

Dr. Michel Bouchon
I.R.I.G.M.-B.P. 68
38402 St. Martin D'Heres
Cedex, FRANCE

Dr. Michel Campillo
Observatoire de Grenoble
I.R.I.G.M.-B.P. 53
38041 Grenoble, FRANCE

Dr. Kin Yip Chun
Geophysics Division
Physics Department
University of Toronto
Ontario, CANADA

Prof. Hans-Peter Harjes
Institute for Geophysic
Ruhr University/Bochum
P.O. Box 102148
4630 Bochum 1, GERMANY

Prof. Eystein Husebye
NTNF/NORSAR
P.O. Box 51
N-2007 Kjeller, NORWAY

David Jepsen
Acting Head, Nuclear Monitoring Section
Bureau of Mineral Resources
Geology and Geophysics
G.P.O. Box 378, Canberra, AUSTRALIA

Ms. Eva Johannisson
Senior Research Officer
FOA
S-172 90 Sundbyberg, SWEDEN

Dr. Peter Marshall
Procurement Executive
Ministry of Defense
Blacknest, Brimpton
Reading FG7-FRS, UNITED KINGDOM

Dr. Bernard Massinon, Dr. Pierre Mechler
Societe Radiomana
27 rue Claude Bernard
75005 Paris, FRANCE (2 Copies)

Dr. Svein Mykkeltveit
NTNT/NORSAR
P.O. Box 51
N-2007 Kjeller, NORWAY (3 Copies)

Prof. Keith Priestley
University of Cambridge
Bullard Labs, Dept. of Earth Sciences
Madingley Rise, Madingley Road
Cambridge CB3 OEZ, ENGLAND

Dr. Jorg Schlittenhardt
Federal Institute for Geosciences & Nat'l Res.
Postfach 510153
D-30631 Hannover , GERMANY

Dr. Johannes Schweitzer
Institute of Geophysics
Ruhr University/Bochum
P.O. Box 1102148
4360 Bochum 1, GERMANY

Trust & Verify
VERTIC
Carrara House
20 Embankment Place
London WC2N 6NN, ENGLAND